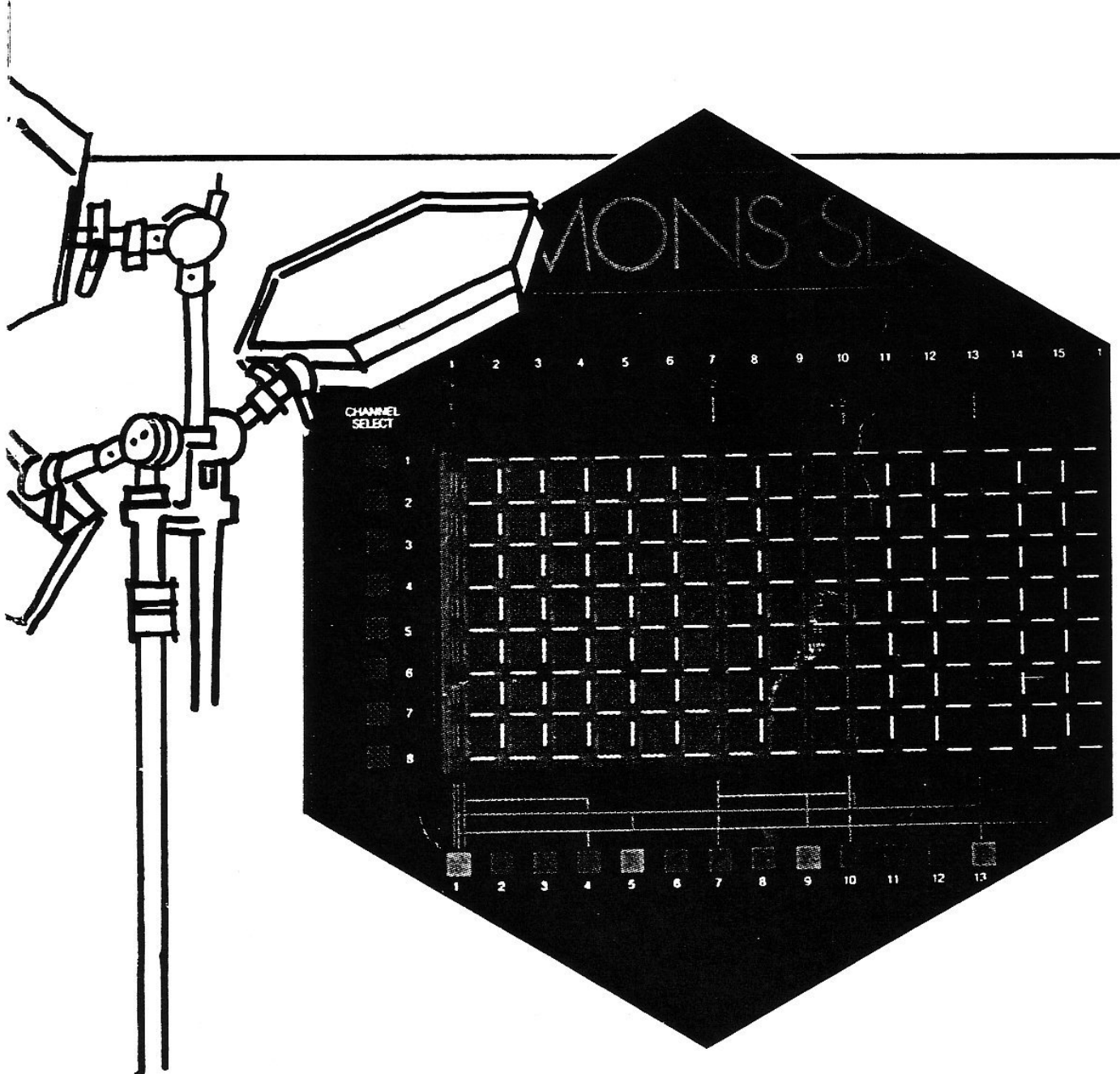


SIMMONS



SDS6 manual

INDEX

	CHAPTER	PAGE
Introduction	1	1
Connecting Up	2	2
Getting Started	3.1	4
Switching On	3.2	6
SECTION 1. Basic use of the SDS 6		
Program A Pattern	4.1	7
Store A Pattern	4.2	9
Play A Pattern	4.3	10
Program A Sequence	5.1	12
Play A Sequence	5.2	13
Undefined	5.3	14
Show	6	15
SECTION 2. Advanced use of the SDS 6		
More Complicated Patterns	7.1	18
Loop	7.1	18
Start	7.3	20
Ext. Short Pattern	7.4	21
Tempo	7.5	23
Dynamics	7.6	24
Clear	7.7	28
Hi Hat and Channel Eight	8	29
More Complicated Sequences	9.1	31
Sequence Editing	9.2	32
Stop	9.3	34
Songs	10	36
SECTION 3. Tape Sync. Gates. Footswitches. Humanizer		
Humanizer	11	37
Footswitches	12	38
Gates In And Out	13.1	39
Tape Sync	13.2	40
Expansion Socket and Memory Dump	14	41
SECTION 4. Quick Programing List		
Program Patterns	15.1	42
Program Sequences	15.2	43
Program Songs	15.2	43
Play Back Patterns	16.1	44
Play Back Sequences	16.2	45
Play Back Songs	16.2	45
Show	17	46
Clear & Misc.	18	47
APPENDICES		
Trouble Shooting	19.1	48
Error Messages	19.1	48
What The Hell Was That?	19.2	49

INTRODUCTION

The SDS 6 is a fully programmable 8 channel sequencer designed for use with the SDS 5 modular drum synthesizer. It has been designed to allow the non-drummer and composer access to the exciting sounds of the SDS 5 as well as widening the horizons of drummers and percussionists who wish to play 'live' (using SDS 5 drum pads) over composed passages stored in the SDS 6 sequencer. Although great stress was laid on the ease of programming during the design of the SDS 6 for musician and non musician alike, it is strongly recommended that this manual is worked through from start to finish. There are no 'special' programming languages or tricks to be learnt. You will be playing music from the start and by the end of the manual you will have a thorough knowledge of the wide and exciting possibilities offered by the SDS 6.

UNPACKING

When you unpack the SDS 6 inspect the unit carefully for damage

You should have included with the SDS 6:

- 8 coded **Jack** — **Jack** leads in cable form
- 1 **Mains Lead** for connection to your country's domestic mains supply
- 1 **Cannon** — **Cannon** lead. For use with the SDS 5 Hi-hat
- 1 **Guarantee Card** Please fill in and return
- 1 **Start Footswitch**
- 1 **Stop Footswitch**
- This User's Manual
- 1 **SDS6 Pattern Notepad**

CONNECTING UP

Connect the SDS 6 to the SDS 5 by plugging the jack to jack leads supplied into the SDS 6 trigger outputs and the other ends into the SDS 5 synth trigger inputs.

All channels are identical on the SDS 6. i.e. Any channel on the SDS 6 can trigger any channel on the SDS 5. But Channel 6 and Channel 8 have special features which increase the flexibility of the SDS 6 when used with cymbals and hi-hats.

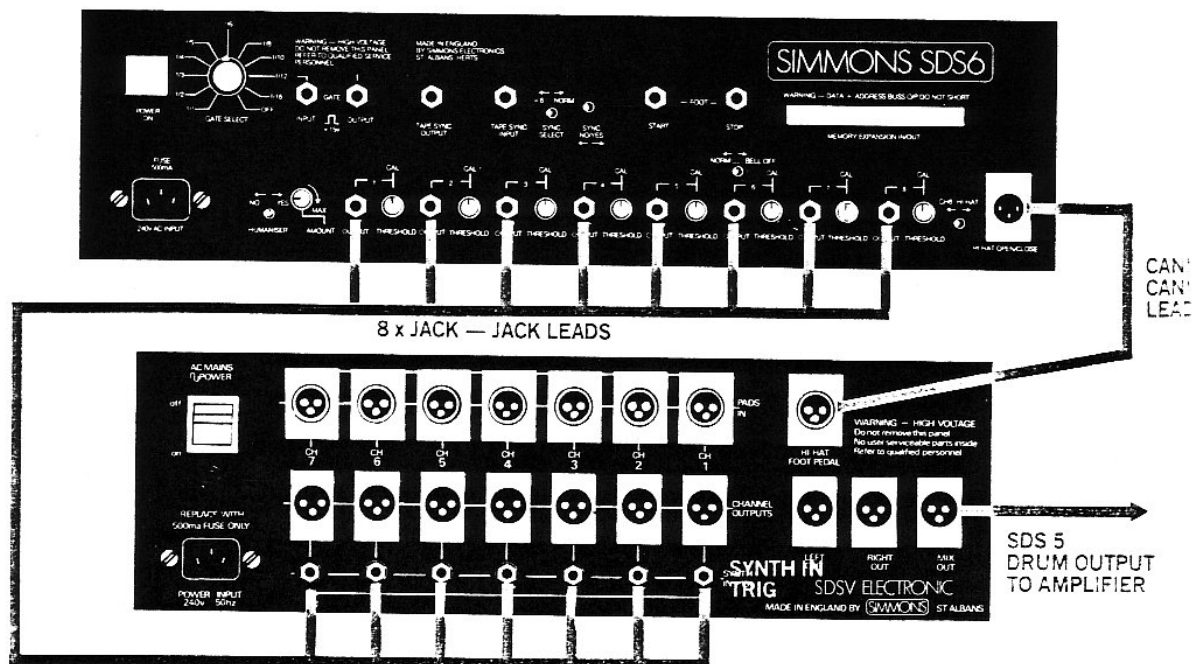
If you have a hi-hat in the SDS 5 rack use the cannon to cannon to connect Channel 8 of the sequencer to the Hi-hat pedal input of the SDS 5. (This allows the special Channel 8 on the sequencer to open and close the SDS 5 Hi-hat). (See Chapter 8).

If you have a cymbal in the SDS 5 rack use Channel 6 of the sequencer to trigger it. (Channel 6 has the 'bell' on/off switch on the rear panel).

Of course once you have mastered the SDS 6 you can try swapping leads around. Patterns programmed for bass, snare, hi-hat and tom toms can sound amazing when the leads are swapped so that the bass pattern hits the snare drum, the snare pattern hits the hi-hat etc.

This manual assumes

- Channel 1 is connected to bass drum**
- Channel 2 is connected to snare drum**
- Channel 3 is connected to hi tom**
- Channel 4 is connected to med tom**
- Channel 5 is connected to low tom**
- Channel 6 is connected to cymbal**
- Channel 7 is connected to hi hat trigger**
- Channel 8 is connected to hi hat open/close**



SDS 5
DRUM OUTPUT
TO AMPLIFIER

Connect the SDS 5 to an amplifier or mixing desk as described in the SDS 5 manual. (Use mix outputs, stereo outputs or individual outputs).

The standard SDS 5 pads, or suitcase pads can be plugged in as well as the sequencer triggers and can be used to play along with the sequencer.

Check the following back panel switches.

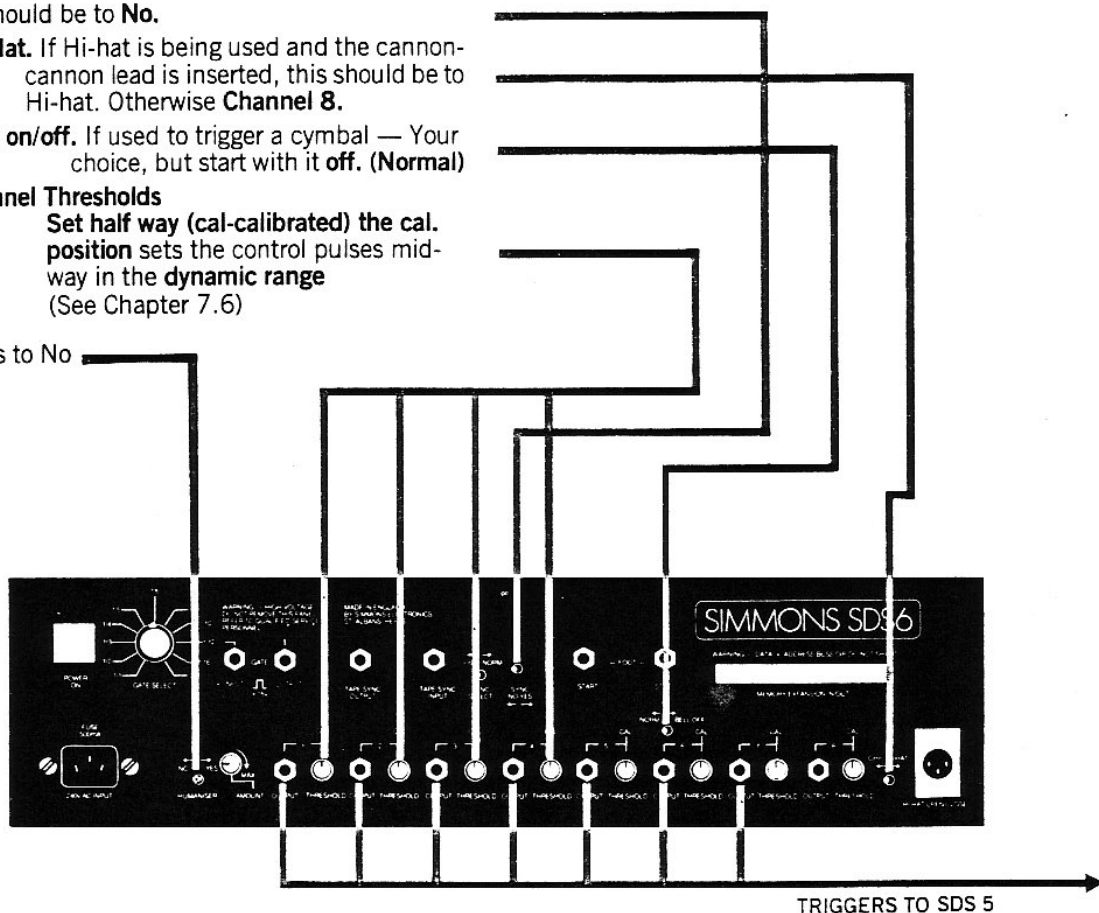
Sync — Yes/No should be to No.

Channel 8 — Hi Hat. If Hi-hat is being used and the cannon-cannon lead is inserted, this should be to Hi-hat. Otherwise Channel 8.

Channel 7 — Bell on/off. If used to trigger a cymbal — Your choice, but start with it off. (Normal)

Channel 1-8 Channel Thresholds
 Set half way (cal-calibrated) the cal. position sets the control pulses mid-way in the dynamic range (See Chapter 7.6)

Humaniser No Yes to No



GETTING STARTED

The programming instructions in this manual are in three sections

Section 1 — Basic use of the SDS 6, programming simple patterns and sequences. Storing them and playing them back.

Section 2 — More advanced patterns, use of dynamics, short patterns, extended patterns, editing sequences, advanced sequence programming, programming songs.

Section 3 — Gates in and out. Tape sync and footswitches. Humanizer.

Section 4 — Quick programming list. For use once you are proficient in programming the SDS 6. This part of the manual lists in "button pushing" terms all the functions of the SDS 6.

You must be itching to start by now!

Well, before you do, please read this brief description of the programming concepts behind the design of the SDS 6.

1. The matrix is a 32 x 8 (Channel) light emitting diode (LED) display, which displays each programmed pattern of drum music. Each channel corresponds to a particular drum (you choose which) connecting the SDS 6 to SDS 5 drum modules). If a light is on, a drum will be hit when the scan line passes it. The matrix is the heart of the SDS 6. It allows you to see (and optionally change) instantly all the 'hit drum' information as it is being played.

2. Hits are entered into the matrix by selecting the Channel (1 to 8) by pressing one of the 8 buttons running down the left hand side of the matrix and then, whilst holding this button, pressing one of 32 'position' buttons which run along the bottom of the matrix. Where the position and channel lines cross an LED will glow, indicating a drum will be 'hit' at this position. Selecting the position again will switch the LED off.

3. The scan line shows the progress of 'time' through the matrix the faster the scan line moves the faster the tempo.

4. A **pattern** is a matrix (32 x 8) with 'hits' stored in that matrix. You assign each pattern a number (between 1 and 99). Each matrix of information can be recalled by simply asking for a particular pattern e.g. Play pattern 1, play pattern 53, play pattern 19 etc.

5. You assign each sequence a number (between 1 and 99) and each sequence can be recalled by asking for a sequence number, e.g. play sequence 3, play sequence 11 etc. Each sequence can consist of up to 250 patterns.

6. You assign each song a number, and as with patterns and sequences, the songs can be recalled by asking for that song number, e.g. play song 99, play song 8, play song 17 etc.

So — to recap:

The basic building blocks of each composition are the 99 patterns you can program.

You program each pattern by switching 'hit drum' lights on and off in the matrix.

You can then string these patterns together forming choruses or verses, fills, intros, — forming sequences. Each of these sequences has a number from 1 to 99 and can contain up to 250 patterns.

You can then string these sequences together to form a 'song'. Each song has a number from 1 to 99 and can contain 250 sequences.

A **sequence** is a series of **patterns** played sequentially.

A **song** is a series of **sequences** played sequentially.

Programming patterns, sequences and songs is done in **English** via specific function keys.

Each subsequent step in programming is indicated by a green light glowing above the function keys.

Each previous step is indicated by a red light glowing above the function keys.

Thus your options are displayed in **green**. What you have already done is displayed in **red**.

Important read this before switching on

CONNECTING UP
Europe mains voltage

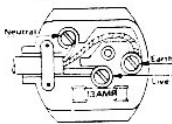
Connect the mains plug according to the following colour code:

Brown — Live

Blue — Neutral

Green/Yellow — Ground or Earth

It is imperative that the SDS 6 is earthed



Check that the voltage label on the back panel matches your domestic mains supply before switching on:

240V — G.B. and Australia

220V — Europe

115V — Canada and U.S.A.

The SDS 6 contains non-volatile memory in which your programs are stored.

These programs can be destroyed if the following simple precautions are not taken.

The SDS 6 must be earthed.

Switch on the SDS 6 last, after other ancillary equipment.

Switch off the SDS 6 first, before other ancillary equipment.

Do not use in an environment where heavy electrical machinery is being switched on and off, or where inductive spikes are present on the mains supply.

Avoid disconnection of the mains during 'program'.

SWITCHING ON

Have you read the previous page?

Switch on ancillary equipment

Switch on SDS 5

Switch on SDS 6

Eight green lights should glow on the SDS 6:

Program. Play. Clear. Store. Show. Tempo. Dump. Load.

As well as an orange light:

Abort

If nothing happens **switch off**

Check all the connections described in Chapters 2-3.2

Try again.

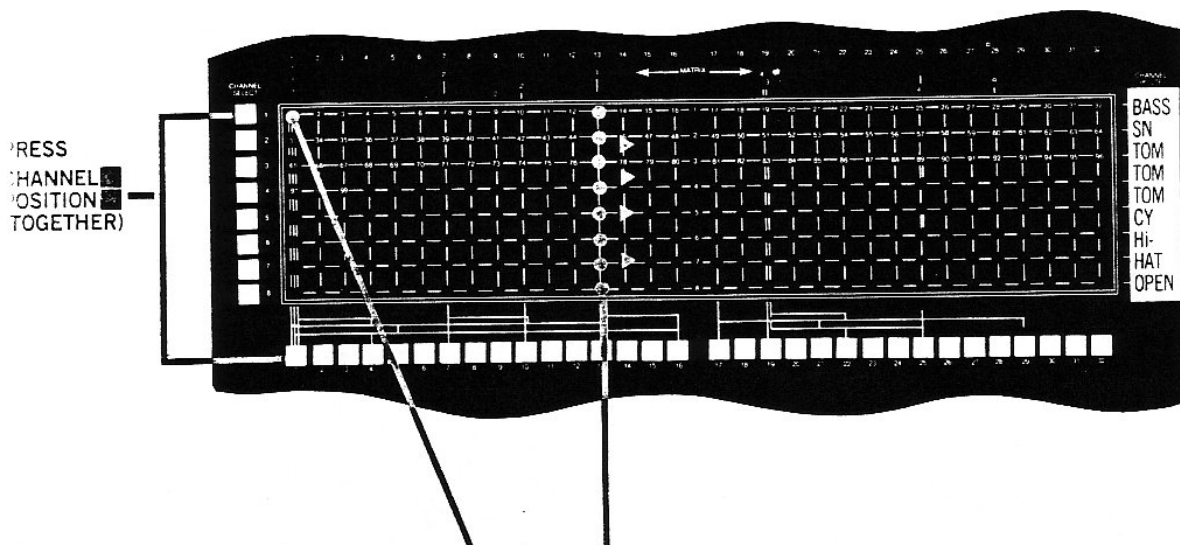
If nothing lights the second time refer to
Chapter 19 **Trouble-shooting.**

PROGRAM A PATTERN AND STORE IT AWAY

Assuming:

- Channel 1 Bass Drum
- 2 Snare
- 3 Hi Tom
- 4 Med Tom
- 5 Low Tom
- 6 Cymbal
- 7 Hi Hat Hit
- 8 Hi Hat. Open/Close

Press Program Pattern



Channel 1 Position 1 will light in the matrix. The scan line will start and proceed across the matrix.

As the scan line passes position 1 the bass drum (Ch. 1) is hit

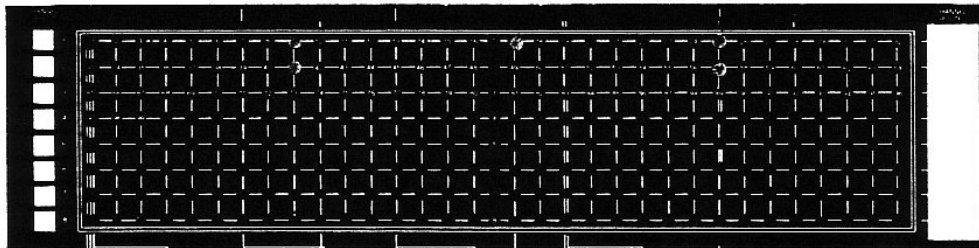
Enter more 'hits' into the matrix by pressing:

- Channel 1 and position 2
- Channel 1 and Position 3
- Channel 1 and Position 25

Enter 'hits' into Channel 2 (Share) by pressing:

- Channel 2 and Position 2
- Channel 2 and Position 3

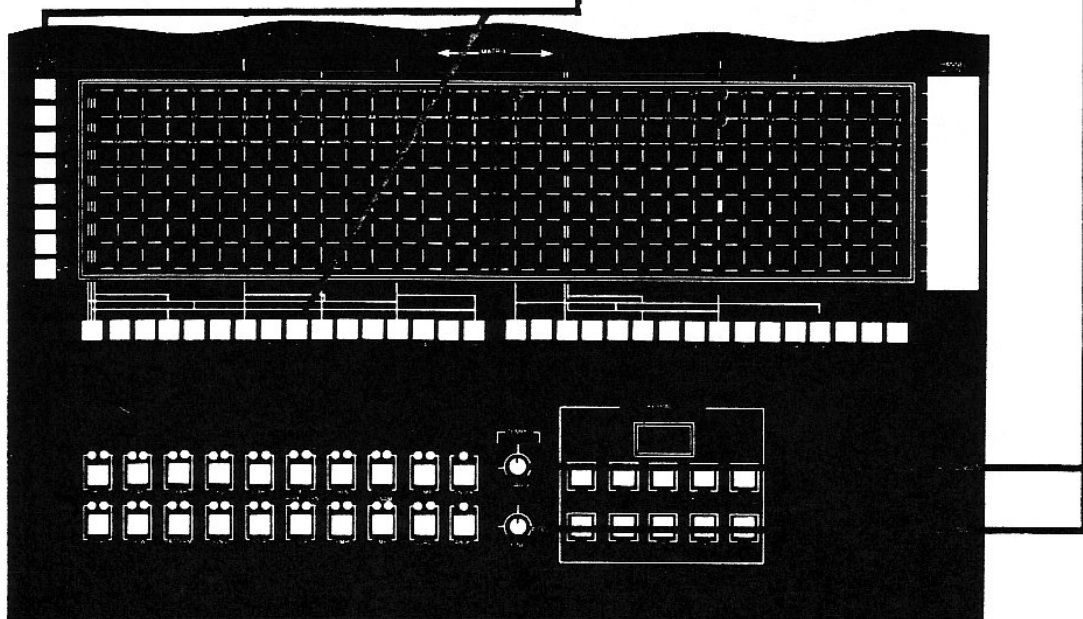
The matrix should look as follows:



Speed up the tempo by altering the positions of the fine and coarse speed controls.

If you make a mistake in entering 'hits' into the matrix, they can be removed by pressing the same channel and position buttons.

Play around with the settings and the SDS 5 bass and snare channels to achieve the desired sounds.



Let's store this pattern before we do anything else.

STORE PATTERN

The memory in the sequencer is where information entered in the matrix is stored away.

Information stored in the matrix is stored away in 32 x 8 blocks. Each block has a unique number, from 1 to 99. These numbers are designated 'pattern' numbers, although each pattern may contain one or more bars of music.

These patterns are not stored away until a specific series of buttons are pressed. This will make you think before valuable data is overwritten in memory.

For example: If a pattern is stored as Pattern 1, any pattern previously stored as pattern 1 is overwritten with the new Pattern 1. The old Pattern 1 is lost forever.

Let's store the pattern you have programmed as **Pattern No. 1**

Press **Store Pattern Keypad** **Enter**

Once **enter** is pressed, the pattern is stored away as **pattern 1**

The pattern number is displayed in the keypad display, and the pattern 'hits' are left in the matrix. So you can add to it or edit it and store as a new pattern.

Let's try that

Remove the bass drum hits at 17 and 25

By pressing

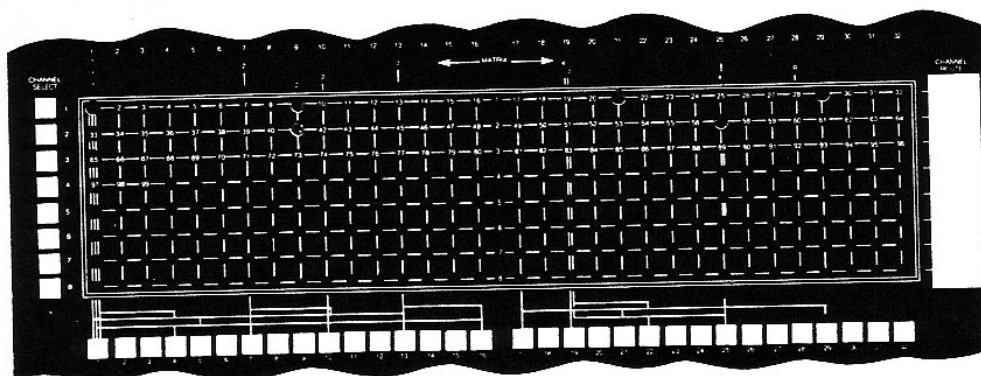
Channel **and Position**
Channel **and Position**

Enter **new** hits at **position 21 and 29**

By pressing

Channel **and Position**
Channel **and Position**

The matrix should look as follows:



Let's store this new pattern as Pattern No. 2
Press **Store Pattern Keypad** **Enter**

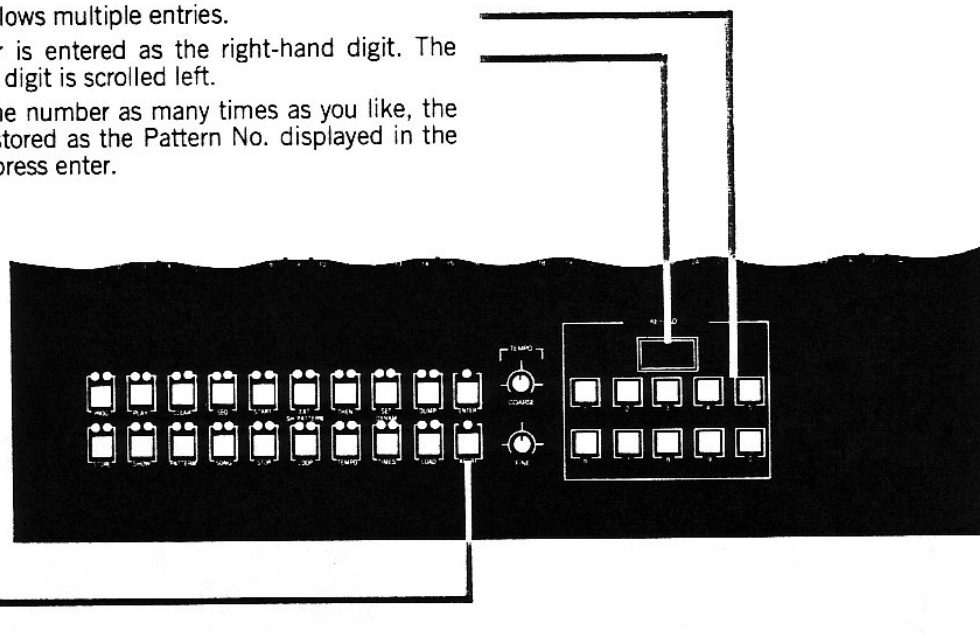
Try entering more information in the Matrix.

Store the new pattern as Pattern 10

Note the keypad allows multiple entries.

Each new number is entered as the right-hand digit. The existing right hand digit is scrolled left.

You can change the number as many times as you like, the pattern is always stored as the Pattern No. displayed in the keypad when you press enter.



You can press **abort** at any time. It returns you to the 'switch on' state. Any information displayed in the matrix which has not been stored will be lost.

Press Abort

Note — You are returned to the 'switch on' state with the initial choice green lights on.

PLAY

You should have 3 patterns in the memory

Pattern 1, Pattern 2, Pattern 10

to play any of these patterns you enter **play** as follows:

Press **Play Pattern Keypad** 

To play the pattern once

Press **Start**

You can change the pattern number being played by entering the new number in the keypad.

Press **Keypad 02 Start**

Pattern 2 will be played

Press **Keypad 03 Start**

Pattern 3 will be played

To play the pattern over and over

Press **Loop**

Pattern 1 will be retrieved from memory, displayed on the matrix and played at the tempo set by the tempo controls.

(The **loop** key is a multi-function key that is used to shorten the matrix length — Chapter 7.2 — as well as allowing patterns and sequences to be played continuously)

Be careful that you have the correct number in the keypad display before you press **start**. It is easy to have previous entries scrolled left e.g. 1st entry 2. 2nd entry 3 — result = 23, not 3 — enter 03 for 3. The result of attempting to play a non-existent pattern will be the error message **UN** (undefined see Chapter 5.3).

New numbers can be entered whilst a pattern is **looping**. The new number is displayed as **II** it is entered but is overwritten when the scan line reaches the end of the matrix by the pattern number that is being played.

The new pattern is not played until you press **start**. When the first pattern has finished playing the new pattern will be played.

Press **Abort**

To Play Pattern 1 continuously

Press **Play Pattern Loop Start**

To Play Pattern 2

Press **02 Start**

To Play Pattern 10

Press **10 Start**

So you can see it's easy to see how patterns sound played one after the other by entering pattern numbers manually.

Read on to see how its done automatically.

PROGRAMMING SEQUENCES

Sequences are a series of patterns played consecutively. You would normally arrange sequences to represent choruses, verses, intro's etc. so that when you compose a song (chapter 10) it's easy to shuffle the verses and choruses around, as each is represented by a sequence number.

Supposing we wanted to play **Pattern 1** followed by **Pattern 2** followed by **Pattern 10**?

We can do this by stringing the patterns together as a **sequence** (of Patterns)

Assuming that the sequencer is in the **switch on** state (Press **abort** if it is not)

Enter the program sequence mode as follows:

Press **Program Sequence**

You can then string patterns together to form a sequence as follows:

Press **Play** **Pattern 1** **Pattern 2** **Pattern 10**

You can then store the sequence. You have to give each sequence a number (from 1 to 99) so it can be recalled later.

Store this sequence as number 1.

Press **Store Sequence 1** **Enter**

You can **abort** at any time before you press **enter** and the main memory is left unaltered. The sequence is stored when you press **enter**.

Any sequence that was previously given the number 1 is lost -- the new sequence 1 has overwritten it. So be careful to allocate a new sequence number if you want to keep old sequences intact. (see **show** Chapter 6 to see free sequence number.)

The tempo set on the tempo controls is also stored with the sequence. When the sequence is played, it will be played at the same tempo as was set when it was programmed.

You can store 99 sequences

PLAYING SEQUENCES

Assuming that the sequencer is in the **switch on** state (Press **abort** if it is not)

To play a stored sequence e.g. sequence 1:

Press **Play Sequence** **[1]**

To play the sequence once:

Press **Start**

To play the sequence over and over:

Press **Loop**

Sequence 1 will be retrieved from memory

Pattern 1 will be displayed and played

Followed by **Pattern 2** and **Pattern 10**

The **pattern number** presently being played is displayed on the keypad display.

You can stop the sequence at any time by pressing the **stop** button. The pattern currently being played is finished before the sequencer stops. To start the sequencer again press **start**. The sequence starts again from where it stopped.

While the sequence is playing try moving the **tempo** controls. Nothing happens, the tempo for the sequence is retrieved from memory — it was stored there when the sequence was programmed.

This tempo can be overridden during play by starting the sequence as follows:

(Abort first)

Press **Play Sequence Tempo** **[1]** **Enter** **Loop** (or start)

Now while the sequence is playing, it gets its tempo from the front panel **tempo** controls — overriding the stored tempo value.

You can stop a sequence four ways.

- 1 It will stop by itself at the end of the sequence if not looped.
- 2 It will stop if encounters an **SP** (Stop) instruction in a pattern (See Chapter 9.3).
- 3 **Press stop** — The sequence will stop at the end of the pattern presently being played. Pressing **start** re-starts the sequence where it left off.
- 4 **Press abort** — Stops the sequence immediately and returns to **switch on** condition.

UNDEFINED

If so far you have made no mistakes, then you will not have seen any error messages.

There are various error messages which are listed in Chapter 19. However the undefined pattern or sequence is one you should know immediately, if you have not already come across it.

If during program **pattern** you had inadvertently stored a pattern as number 20 instead of number 2 and then attempted to play **pattern 2**, the error message **UN (Undefined)** would be displayed in the keyswitch display.

Let's try playing a pattern that does not exist.

Press Play Pattern 5 Enter

If there is no **Pattern 5**, **UN** will be displayed along with a green light over **show**

Press Show

A red light glows over **pattern** and the undefined **Pattern No. (5)** is displayed — telling you **Pattern 5** is **UN**defined.

The **UN**defined error will also be displayed if an undefined pattern is entered in a sequence or if an attempt is made to play an undefined sequence.

How do you keep track of all these patterns and sequences?

Read on

SHOW

We just used the **show** button to inform us of the **pattern no.** that was undefined. **Show** can also inform us of how much memory has been used, what pattern, sequence and song numbers have been used and what information is stored in a pattern.

Press **abort** to return to the **switch on** state.

Press **Show**

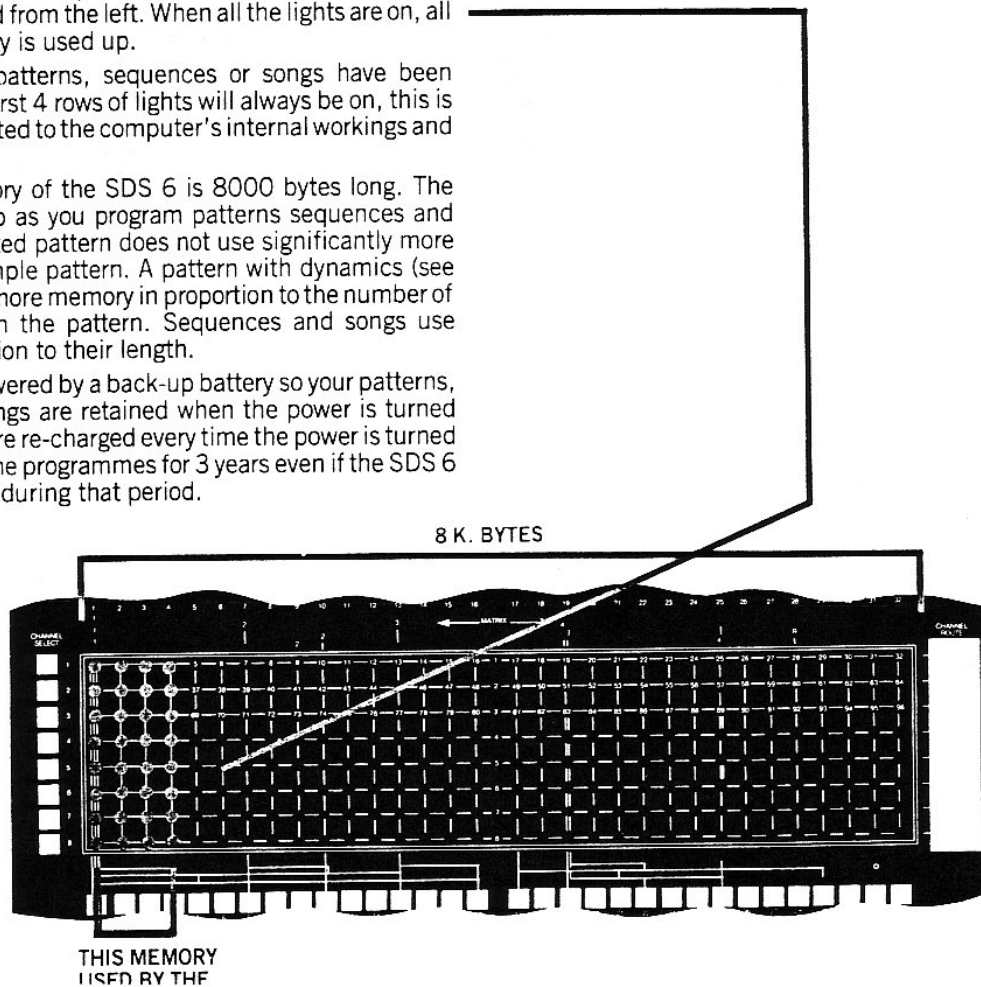
The display is turned into a picture of the sequencer's memory. The memory is filled from the left. When all the lights are on, all the internal memory is used up.

Note: even if no patterns, sequences or songs have been programmed, the first 4 rows of lights will always be on, this is the memory dedicated to the computer's internal workings and is protected.

The internal memory of the SDS 6 is 8000 bytes long. The memory is used up as you program patterns sequences and songs. A complicated pattern does not use significantly more memory than a simple pattern. A pattern with dynamics (see Chapter 7.6) uses more memory in proportion to the number of dynamics stored in the pattern. Sequences and songs use memory in proportion to their length.

This memory is powered by a back-up battery so your patterns, sequences and songs are retained when the power is turned off. The batteries are re-charged every time the power is turned on and will retain the programmes for 3 years even if the SDS 6 is not switched on during that period.

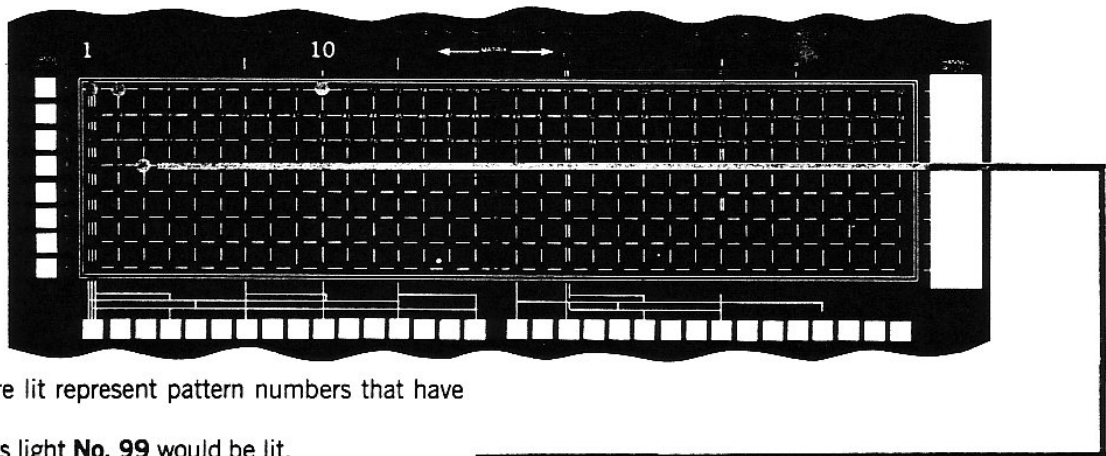
If you require more memory an 8000 byte extension pack is available which plugs into the expansion socket (recommended for back-up see Chapter 14).



If you want to see how many patterns are stored in memory

Press **Pattern**

The matrix should look like this



The LED's that are lit represent pattern numbers that have been used.

If **Pattern 99** exists light **No. 99** would be lit.

If you want to look at the information in a particular pattern, press its number e.g.

Press **2**

Pattern 2 will be displayed.

If you want to hear what **Pattern 2** sounds like

Press **Start**

Pattern 2 will be played once.

Note you can re-start the pattern before it's finished playing by pressing **start** again — very useful for messing about with existing patterns.

To show **Pattern 10**

Press **10**

Press **Start (To Play)**

Press **Abort**

You can also see the sequence numbers that have been stored

Press Show Sequence

The display will show the sequences so far used.

If you wish to see the patterns stored in the sequence (e.g. sequence 1)

Press [1] Enter

The **tempo** value is displayed in the keypad display (this is the tempo set on the tempo controls at the time sequence 1 was programmed)

Press Then

The first Pattern No. in the sequence is displayed

Press Then

The second Pattern No. in the sequence is displayed

Subsequent presses of **then** will display all the patterns in a sequence until the end.

In our original example

Pattern 1, 2, 10 would be displayed.

SUMMARY

So far you should be able to program simple patterns, store them and play them.

Program sequences, store them, play them and show where patterns and sequences are stored in memory.

Summary of Button Pushing

Program Pattern

Program Sequence

Play Pattern NN Pattern NN Pattern NN Pattern NN....

Store Pattern NN Enter

Store Sequence NN Enter

Play Pattern NN Start (or Loop Start)

Play Sequence NN Start (or Loop)

Play Sequence Tempo Start (or Loop)

Show Pattern NN (Start)

Show Sequence NN Then (Tempo)

Then (1st Pattern) Then (2nd Pattern)

MORE COMPLICATED PATTERNS

Re-programming existing patterns

An existing pattern can be re-called from memory for re-programming (i.e. Hits can be added, or taken away etc).

Press Program Pattern 1 Enter

Pattern 1 is retrieved from memory and displayed in the matrix, the choice lights encountered in program pattern (Chapter 4.1) are lit.

Choice

Loop	(This chapter)
Short Pattern Ext	"
Start	"
Tempo	"
Store	"
Dynamics	"
Clear	"

Pattern 1 is displayed and played.

Enter more 'hits' in the matrix.

The new pattern can be stored in memory under a new pattern no. For example Pattern 20

Press Store Pattern 20 Enter

The edited version of Pattern 1 is stored as Pattern 20.

You can of course, store the edited pattern back as pattern 1, overwriting the old pattern 1.

Press Store Pattern 1 Enter

The edited version of Pattern 1 is left on the matrix so that you can make more changes if you wish, storing subsequent versions of pattern 1 under new pattern numbers.

This is the normal way to create a "library" of patterns — adding extra snare 'hits' or tom fills to existing patterns and storing them as new patterns.

LOOP

The normal matrix length is 32 — This can be shortened to any length by selecting loop.

You can then put a line in the matrix, the scan line cannot pass this line and the pattern is 'looped' at this position.

Press **Abort**

Press **Program Pattern 11 Enter**

Press **Loop**

Notice the pattern scan stops, no green lights are on.

Choose where you want the pattern to end by selecting any Channel (1-8) and pressing a position button (1-32) on the matrix.

Try position 17.

A line will be written into the pattern at position 17.

The scan line will start but when it reaches position 16 and encounters the loop line it is sent back to position 1.

Press short pattern (again).

You can now enter different positions for the loop line.

If you want to delete the loop line,

Press **Clear**

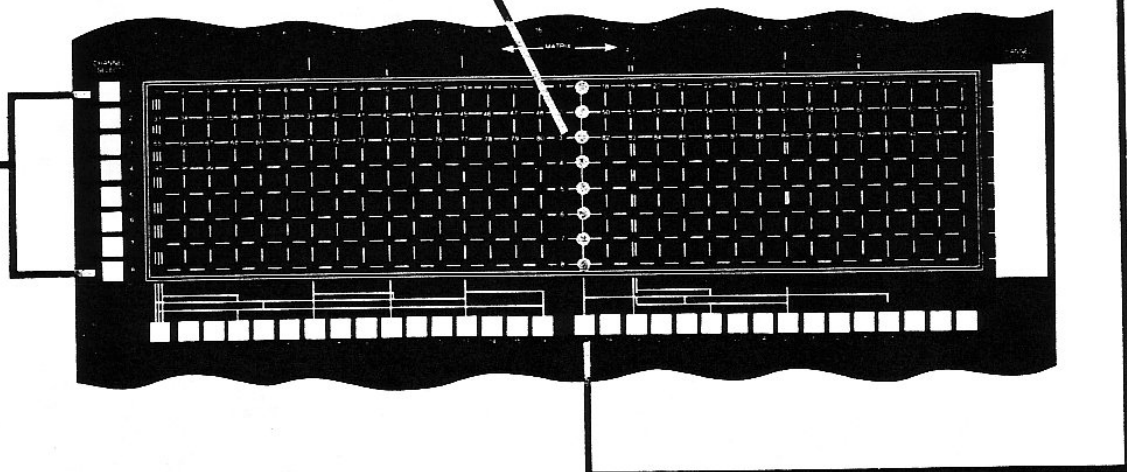
You can move the loop line around as much as you like, experimenting with different pattern lengths.

The pattern line is not stored in main memory until you

Press **Store Pattern 11 Enter**

The shortened-pattern is stored as Pattern 11. Any information that was to the right of the loop line is lost.

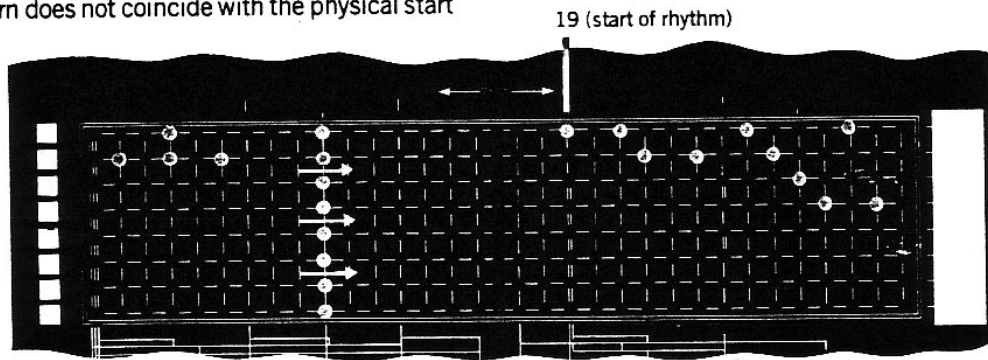
ANY ONE
(I don't care)



START

During composition of a pattern where 'hits' have been entered indiscriminately, a rhythm may become apparent that will fit into an existing or future composition but the start of the rhythm in the pattern does not coincide with the physical start of the matrix.

e.g.



If the pattern is played over and over, a rhythm is apparent with beat No. 1 at Position 19.

If this pattern is used in a sequence, the rhythm will start half way through the pattern.

To correct this, you could of course re-program the pattern, rotating all the 'hits' to the left until the first beat of the rhythm coincides with position 1 of the matrix. — A laborious task.

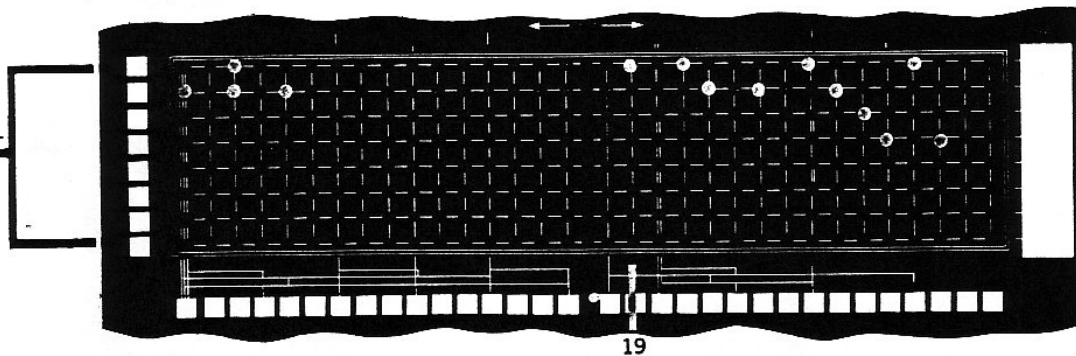
During program pattern, the start button can be used to rotate the 'hits' in the pattern.

All you do is (Assuming you're in program pattern)

Press Start

The scan stops. All green lights disappear and you 'point' at the position that you want to be at the start of the matrix by selecting a channel (1 of 8) and a position.

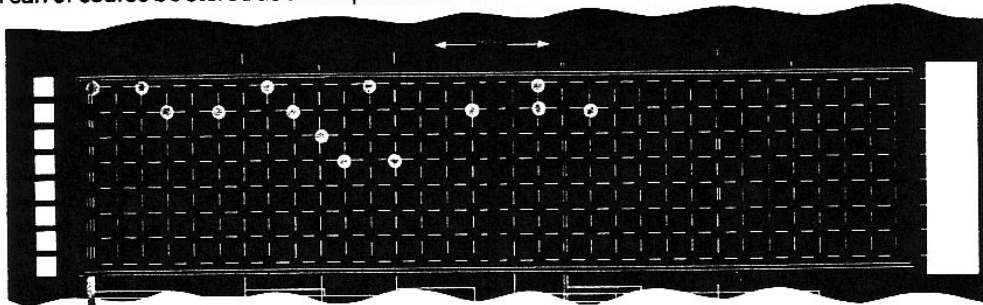
ANY CHANNEL
(Don't care)



The 'hits' in the pattern will be rotated left so that what was at position 19 is now at the start of the pattern (Position 1).

This rotated pattern can of course be stored as a new pattern in memory.


(Note — any dynamic values stored with the pattern will be lost during the rotation of the bar, the dynamics in the original bar are of course safe.)




EXT. SHORT PATTERN (EXTENDED)

In the above example, the scan line moved through the pattern at the same speed as it would if it had to play all 32 beats, the loop pattern line sending the scan back to position one.

How would you program the following notation?

Bass drum 4/4 

Bass drum 4/4 

The first pattern is easy.

32 is divisible by four, so the whole matrix can be used, each bass drum having 8 'steps' (position 1, 9, 17, 25 snare position 9 and 25).

The second pattern would have to be shortened to a number that is a multiple of 3 and 4. Which is 12 or 24.

If we select 24. The bass will be on positions 1, 7, 13, 19

The snare would be 1, 5, 9, 13, 17, 21

If these two patterns are played consecutively with the scan line running at the same speed, the second pattern would sound as if it sped up. (Because it is shorter).

If however, the second pattern was shortened by using the extended short pattern instead of the loop, the sequencer would automatically adjust the tempo (slow it) so the shortened pattern takes the same time to play as the previous full length pattern.

In short: 24 steps played in the time of 32.

Extended short patterns are primarily used in music that mixes 4/4 with triplets, but it can be used where any two patterns which have been shortened are required to be played in the same time.

Further examples can be studied in the patch book. But let's try the common one.

— consecutive patterns
one 4/4, one 4/4 with triplets.

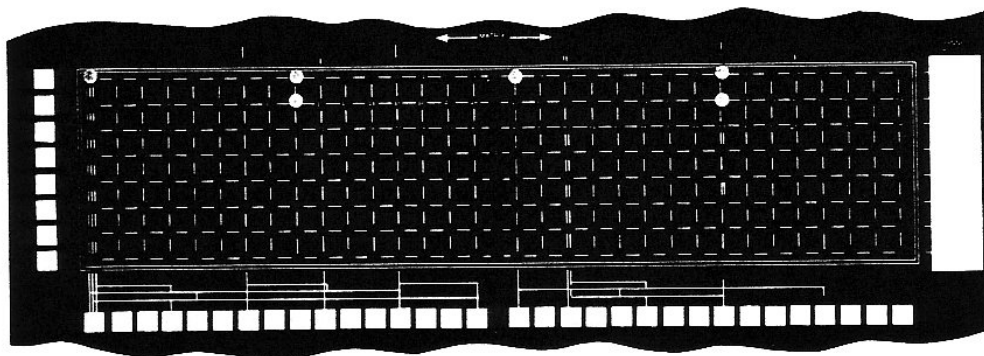
You will have to program patterns and sequences in this example. I'll presume you can do it!!

The total button pushing required is as follows

(Press Abort)

Press Programme Pattern

Enter in matrix

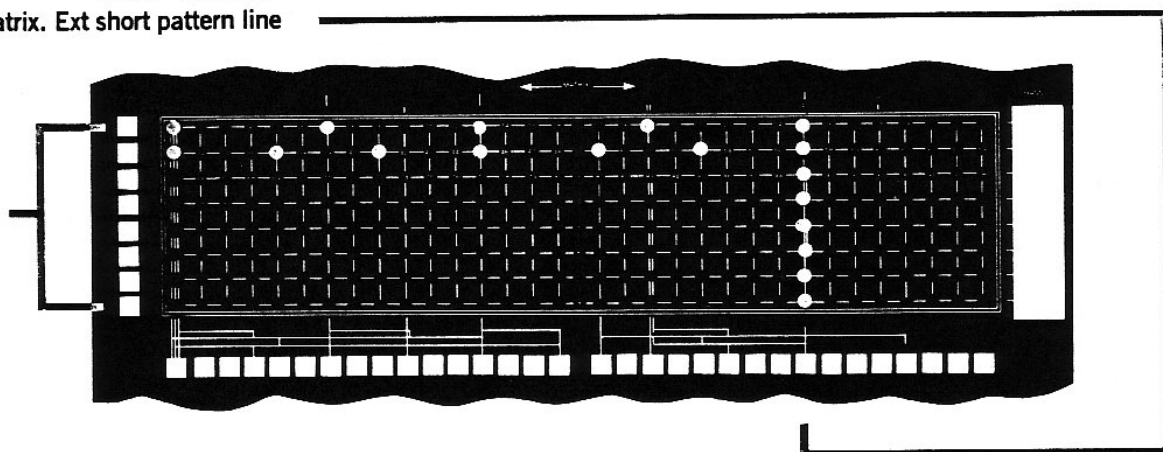


Store Pattern 30 Enter

Press Ext Short Pattern

Enter in matrix. Ext short pattern line

Don't care



Delete bass drum at 9, + 17
Add snare at 1, 5, 9, 13, 17, 21

Add bass drum at 7, 13, 19

Store Pattern 31 Enter

Press Abort

Press Prog Seq Play Pattern 30 Pattern 31

Store Seq 5 Enter

Press Play Seq [] Enter

Press Play Seq [] Start (or loop to play continuously)

Note the scan line slows down during pattern 31, but the bass drum is played at the same tempo.

Try editing pattern 31 and changing the ext short pattern line to a loop pattern line.

Press Play Seq [] Loop (To play continuously)

Now the second pattern is played faster (The scan line moves at the same tempo).

Note also that the red light over ext short pattern (or loop if you've changed it) lights during pattern 31 playback to inform you visually that the pattern is shortened or extended shortened even if it is not apparent aurally.

TEMPO

Pressing the tempo during program pattern (it does different things during program sequence. (Chapter 9.2) doubles the tempo of the scan line.

Press **Abort**

Press **Program Pattern** **Enter**

Press **Tempo**

Note the pattern is played at twice the tempo.

Press **Tempo** (Again)

Note the pattern reverts to normal tempo.

This facility enables two patterns to be strung together (as a sequence) and used where a pattern of 64 'hits' is required.

(The two patterns will take the same time to play as one normal pattern).

The red tempo light comes on during playback to indicate that the pattern's tempo is x 2.

DYNAMICS

This is probably the most important feature of the SDS 6. Up to now all the drums have been 'hit' by the sequencer at the same dynamic level.

The SDS 6 allows you to assign a dynamic level of 1-9 to any individual 'hit' that is displayed in the matrix. 1 is quiet, 9 is loud.

You program dynamics into an existing pattern.

So let's recall a pattern that you have programmed previously.

Press **Program Pattern** **Enter**

Pattern 1 is displayed and played.

All drums displayed in the matrix are hit with the default value of 6. (If you don't assign a dynamic value, all 'hits' are dynamic value 6).

Press **Set Dynamics**

The pattern no. disappears in the keypad display and is replaced by the number 6, which is the dynamic value assigned to all the 'hits' in the pattern (default).

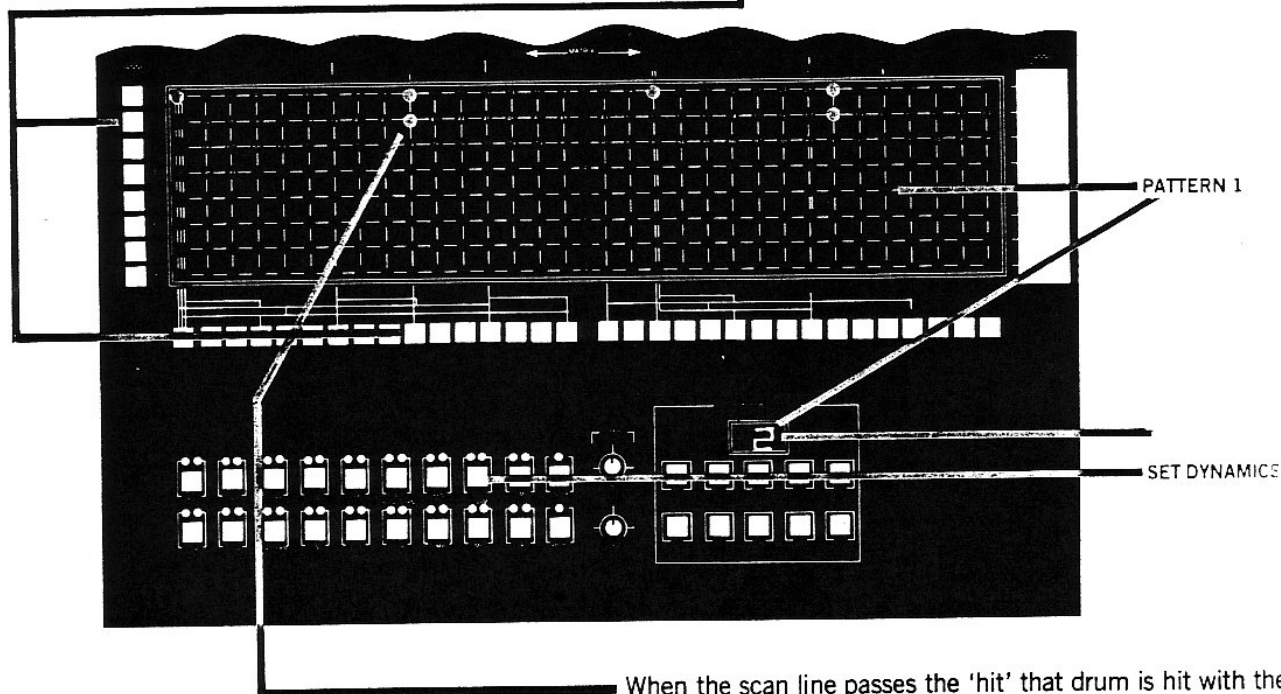
Choose a new dynamic value from 1-9

i.e. 2

Press **2**

The **2** in the keypad display is the current dynamic value which can be assigned to any of the 'hits' displayed in the matrix.

So assign that dynamic value (**2**) to a 'hit' by selecting a channel and position (exactly the same way as 'hits' are entered).

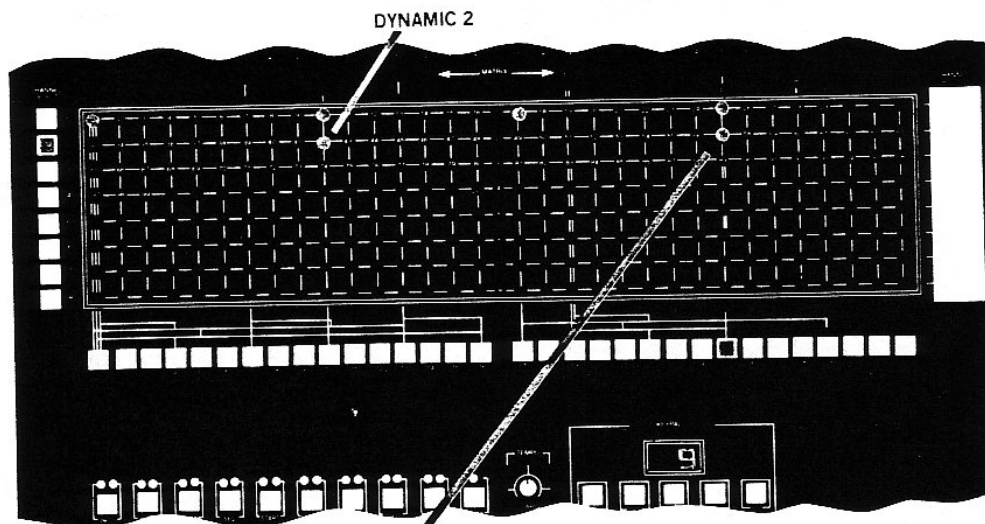


When the scan line passes the 'hit' that drum is hit with the dynamic 2 as opposed to 6, so it sounds quieter.

Choose another dynamic on the keypad

i.e. 9

Press **9**



Assign the value 9 to another 'hit' displayed.
Note that 'hit' is played louder.
You can carry on assigning different dynamics to different 'hits' up to a maximum of 60 different dynamics per pattern.

When you have finished

Press **Enter**

The dynamics are stored with pattern that you are programming but the pattern (with dynamics) is not stored in main memory until you:

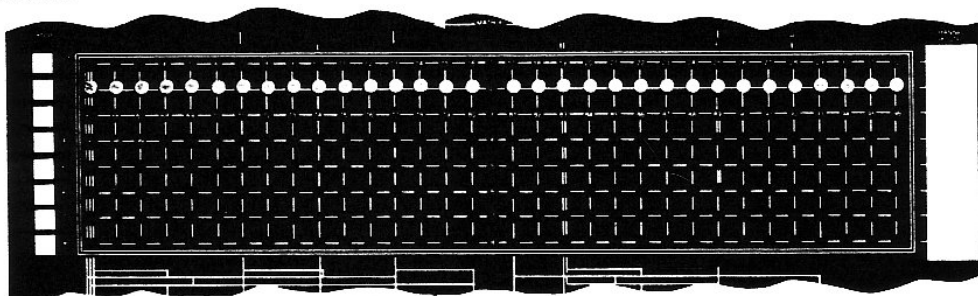
Press **Store Pattern** **99**

The pattern with dynamics is stored as Pattern 99.

The power of the dynamic control of the SDS 6 is best demonstrated on the snare drum.

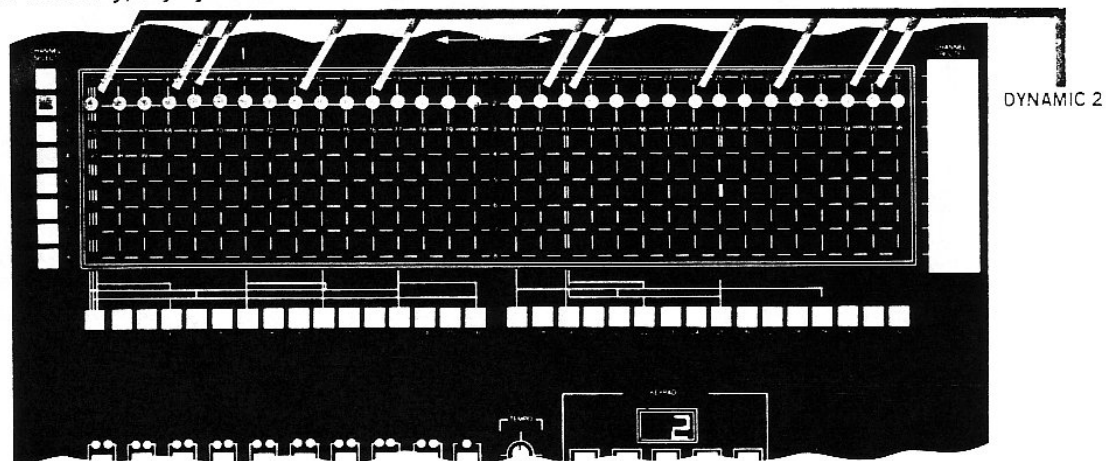
Program a new pattern, entering all 32 'hits' in the matrix.

Press **Program Pattern**

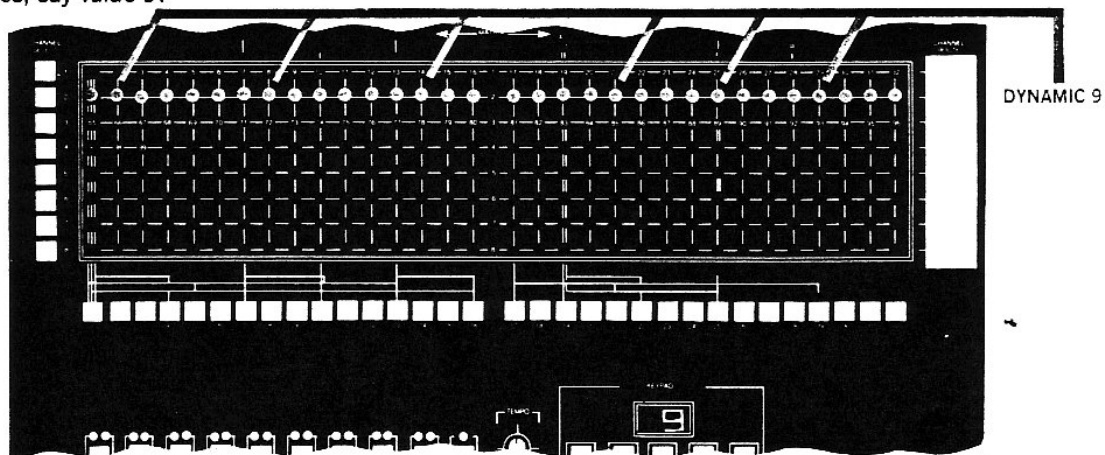


The snare sounds rather like a machine gun, with each 'hit' sounding at the same volume.

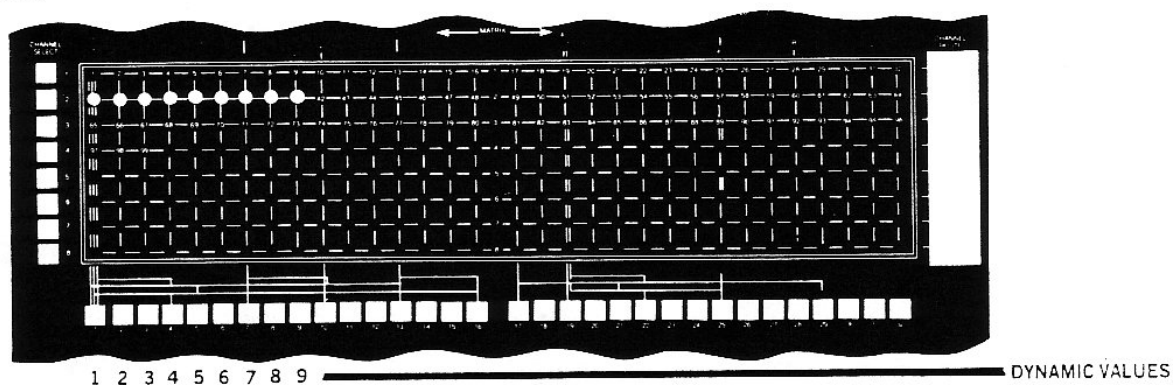
Enter one dynamics randomly, say dynamic value 2.



Add more dynamics, say value 9.



Try entering increasing dynamic values to successive snare hits to create a crescendo.



Whilst this pattern is playing try adjusting the channel 2 (snare) threshold on the rear panel.

At its maximum (fully clockwise) position the dynamics disappear, all hits are played at maximum volume, turn the threshold anticlockwise and the lowest value dynamics disappear.

So, you can set different drums at different dynamic thresholds and it allows dynamic passages to be played without the dynamics (by turning the threshold to maximum) without having to re-program the passage.

NOTES ON MEMORY

It's worth noting that a pattern with dynamics uses far more memory when it is stored away than a normal bar.

So, if you don't need dynamics, don't enter them, a few dynamics can make all the difference but lots of dynamics in all your patterns will use memory very quickly.

CLEAR

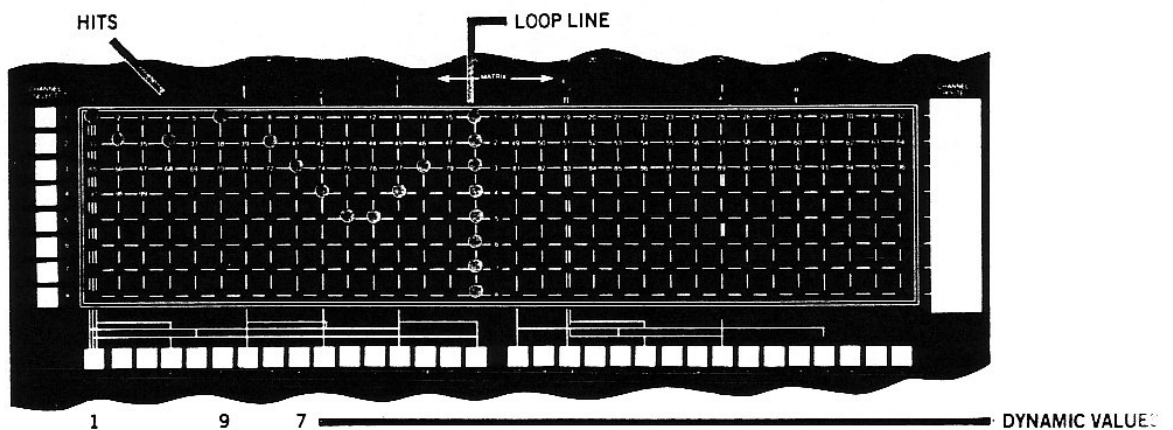
During **program pattern** the green led over the **clear** button is lit, its function is to clear various information from patterns stored in memory.

If the pattern currently displayed in the matrix has only 'hits' (i.e. it's not a short pattern and it does not contain dynamics) then pressing the clear button clears all the 'hit' information in the matrix.

If the pattern contains dynamics these are cleared with the first press of the clear button, the second press clears the 'hit' information.

If the pattern is an extended short bar or short bar, this line is cleared first, followed by any dynamics and the third press clears the 'hit' information.

i.e. successive presses of the clear button clears loop line, dynamics and 'hits' in that order.



HI HAT AND CHANNEL EIGHT

Channel 8 can be used as a standard channel or it can be switched to control the open-close function of the SDS 5 hi-hat (the function normally taken care of by the hi-hat foot switch). When the switch marked CH 8 hi-hat is switched to hi-hat, any hits entered in the matrix in Channel 8 will open the hi-hat. If no hits are entered the hi-hat will be permanently closed.

Hi Hat Open-Close

Channel 8 on the SDS 6 is a special dual function channel. In the normal mode it acts as any of the other channels. It can however be used to open and close the SDS 5 digital Hi Hat module.

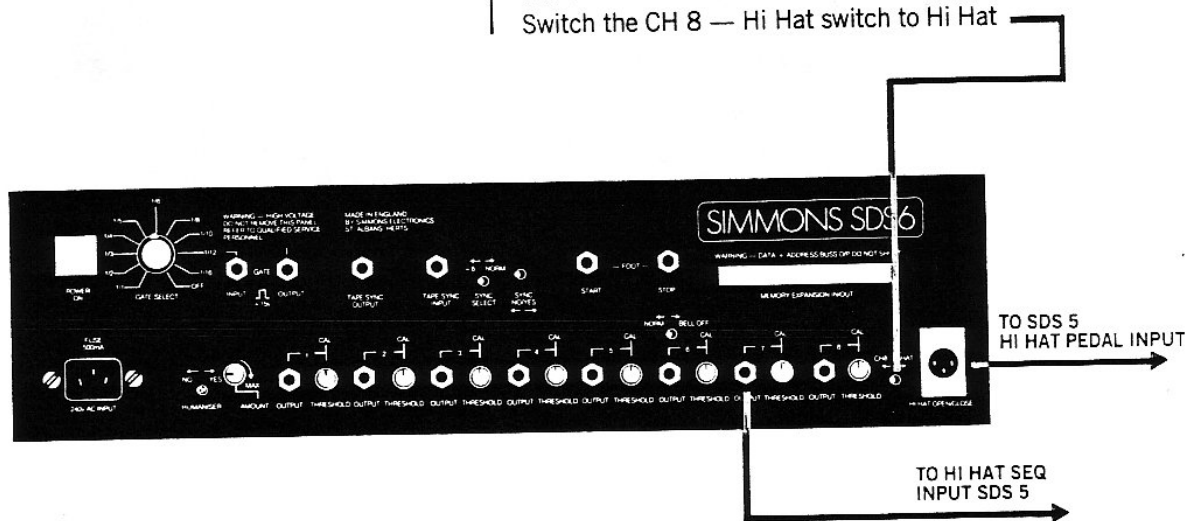
First a brief description of how the Hi Hat works.

It needs 2 signals. One which tells it has been 'hit' and a second which controls whether the cymbal sound is to be long (open or short (closed)).

Normally this is achieved by hitting a pad and depressing or releasing a footpedal. The SDS 6 achieves the necessary control by using 2 channels to control the Hi Hat — one channel to 'hit' the Hi Hats and another Channel (8) to open and close the Hi Hat.

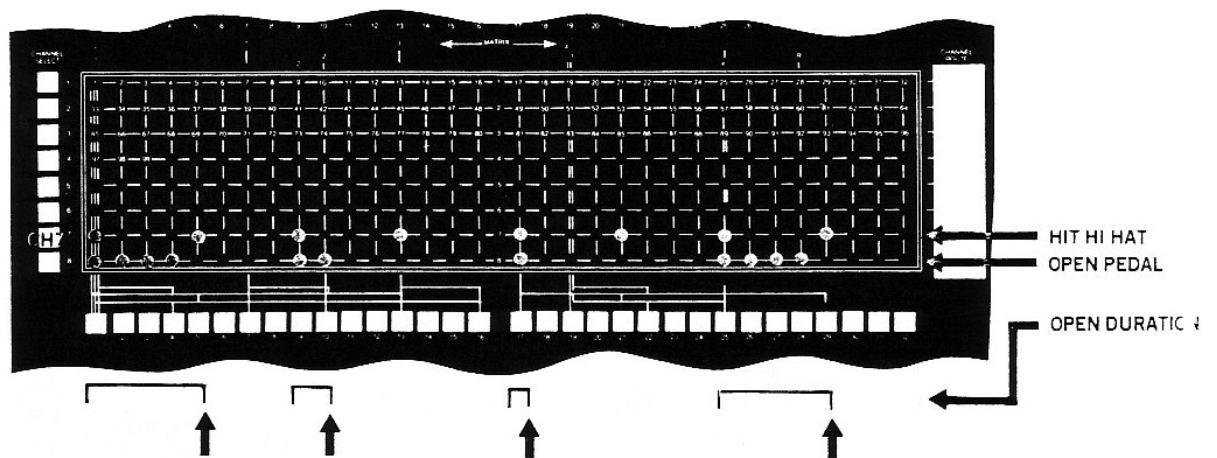
Use channel 7 to 'hit' the Hi Hat and connect CH 8 (cannon output or jack output) to the input marked Hi Hat pedal on the rear of the SDS 5.

Switch the CH 8 — Hi Hat switch to Hi Hat



When CH 8 is used to open/close the Hi Hat, information entered has a different effect. Instead of giving a short pulse for every light in the matrix, CH 8 sends a 'pedal open' signal to the SDS 5. If no lights are on in CH 8 then the Hi Hat is permanently closed.

Set up the following pattern:-



The Hi Hat will be open for the duration indicated by the string of lights in CH 8. It can be seen that the length of the open Hi Hat sound can be varied by the number of lights on in an 'open Hi Hat' command in CH 8.

NOTES:

The maximum length of the open Hi Hat sound is determined by the SDS 5 Hi Hat module, holding the Hi Hat open for 2 seconds with the SDS 6 will not override the maximum delay time (1.5 seconds) of the SDS 5 Hi Hat module.

When the Hi Hat is closed (↑ in above diagram) the Hi Hat is also triggered to give the 'chink' effect of the cymbals closing. Adding dynamics to Hi Hat 'hits' and the open/close commands (add dynamics in the normal way) can have interesting effects as the sound of the Hi Hat will change if the Hi Hats are only half open.

MORE COMPLICATED SEQUENCES

Until now the only way to program a sequence was to enter each pattern individually one after the other e.g.

Play Pattern 3, Pattern 9, Pattern 42, Pattern 78 etc etc.

Most rock music consists of the same pattern repeated many times, entering the patterns in this manner would be very tedious.

The SDS 6 can enter a pattern in a sequence up to 99 times automatically

e.g. Supposing a sequence is to consist of **Pattern 1, Pattern 2** and then **Pattern 6** 50 times.

You could program it as follows:

Program Sequence

Play Pattern 1, Pattern 2, Pattern 6, Pattern 6, Pattern 6, Pattern 6, Pattern 6, Pattern 6 etc.

An easier way is as follows:

Play Pattern 1, Pattern 2, Pattern 6 enter 50 times

Store Sequence 80 Enter

The SDS 6 enters pattern 6 50 times into the sequence.

You are allowed to make these multiple entries as many times as you like in any sequence as long as the total number of patterns does not exceed 250.

You can also make multiple entries of groups of patterns i.e.

(Pattern 40, Pattern 35, Pattern 3) Played 40 Times

e.g. adding this group to the sequence above the entire sequence would be entered as follows:

Play Pattern 1, Pattern 2, Pattern 6 Enter 50 Times, then Pattern 40, Pattern 35, Pattern 3, Enter 40 Times.

The above sequence consists of 172 patterns and was programmed in 24 keystrokes.

The beginning of the group of patterns is marked by the **then** statement, the end of the group by the **enter** statement.

SEQUENCE EDITING

As with editing patterns (Chapter 7.1) where a pattern stored in memory is retrieved so that you can alter it and store it as a new pattern (or store back as the same pattern — over writing the old pattern) you can edit sequences, altering the existing sequence or forming a new sequence from parts of an old sequence.

You enter the edit mode by asking to (re) program an existing sequence.

i.e. Sequence 1 — our original demonstration sequence (see page 12) should still be in the SDS 6 memory (unless you have deleted it — see **show** Chapter 6 to examine the sequence) and should consist of pattern 1, pattern 2, pattern 10.

To edit sequence 1:

Press Program Sequence 1 Enter

Note that the tempo light is lit. This allows you to alter the tempo of sequence 1.

Press Tempo

The new tempo value is saved for the sequence (this is the current tempo set on the fine and coarse tempo controls). You would normally have altered the tempo — trying out different tempos for the sequence, leaving the desired tempo set on the fine and coarse controls.

The last pattern number in the sequence is displayed in the keypad readout — in this case pattern 10.

You can then add patterns to the end of sequence 1 using the notations described in chapters.

At any time you can review your library of patterns without exiting from the program sequence mode.

Just **Press Play Pattern No.**

The pattern will be played, and then you will be returned to the program sequence mode (where you left off).

At any time during program sequence you can play the sequence as it stands at the moment.

Press Start

The sequence you are currently programming will be played, and then you are returned to the program sequence mode.

You can also clear patterns from the end of the sequence — the last pattern in the sequence is always displayed in the read-out and you can always add new pattern numbers to the end of the sequence.

Try clearing patterns from the end of pattern 1.

Press Clear

Pattern number 10 is deleted

Pattern number 2 is displayed which is the next pattern in the sequence.

Press Clear

Pattern number 2 is deleted

Pattern number 1 is displayed which is the next pattern in the sequence.

So now the edited version of sequence 1 contains 1 pattern, but the original sequence 1 still exists in the SDS 6 memory and is not altered unless the new, edited version is stored back as sequence 1.

Add two new patterns to the sequence:

Press Pattern 11 Enter 4 Times Pattern 3

You can of course store the new edited version of sequence 1 as a new sequence, say sequence 20.

Press Store Sequence 20 Enter

You now have 2 sequences made from the original sequence 1. Sequence 1 consists of pattern 1, 2, 10.

Sequence 20 consists of pattern 1, 11, 11, 11, 11, 3, patterns 2 and 10 having been edited by use of **clear**.

Note — If you attempt to play this sequence and pattern 11 does not exist (because you have not written it yet) the error message **UN** (undefined) will be displayed and the sequence will stop (see Chapter 5.3).

STOP

A stop command can be entered into a sequence. When the sequencer encounters this command it stops the sequence, and waits for the start button (or footswitch — see chapter 12) to be pressed, the sequence will then start playing the next pattern in the sequence.

The main use for the stop command is to allow the musician to insert pauses into a sequence, say at the end of an intro or chorus, and re-start the sequence manually in time with some other event, e.g. a guitar intro or riff.

It is also useful when a sequence needs to be played repeatedly with pauses in between.

If sequence 1 is played in the loop mode the cycle pattern 1, 2, 10 1, 2, 10 1, 2, 10 etc is played continuously without a break.

If, however a stop command was inserted at the end of a sequence 1 it would stop at the end of the sequence, but would not return you to the "switch-on" state but wait for the start button to be pressed, whereupon it would start to play the sequence again from the start.

Try editing sequence 1, adding the stop command at the end of the sequence.

Press Program Sequence **Enter Timing**

Press Stop

Note the code **SP** appears in the keypad display.

The stop command is now added to the end of sequence 1.

To up-date the old sequence 1 in memory store the new sequence 1 as sequence 1.

Press Store Sequence **Enter**

The old sequence 1 is overwritten with the new version which has the stop command at the end.

To see the new sequence use show:

Press Show Sequence **Enter**

Shows the tempo value.

Press Then

Shows the first pattern (1)

Press Then

Shows second pattern (2)

Press Then

Shows third pattern (10)

Press Then

Shows **SP — Stop**

Play sequence 1

Press Play Sequence **Loop**

Sequence 1 is played, when it encounters the stop command it stops, displays **SP**. Note the start light is lit.

Press Start

The sequence is played again and stops as before. This series of events is repeated until you press abort.

SUMMARY OF BUTTON PUSHING

Program sequence — enters program new sequence

Play pattern NN enter NM times — enters pattern **NN NM** times in the sequence (**NN x NM**)

Then Pattern NN, Pattern NM, Pattern NO, Pattern NP, Enter NQ Times — enters the series of patterns **NN NM NO NP, NQ** times in the sequence (**NN, NM, NO, NQ**) x **NQ**.

Program sequence NN — retrieves sequence **NN** for editing.

Enter Timing — changes old tempo to new tempo (no change if the sequence was played at its original tempo just previously to editing).

Clear — clears patterns sequentially from the end of the sequence, the last pattern is always displayed in the read-out.

Pattern NN Pattern NM Pattern NO — pattern **NN, NM, NO** are added to the end of the sequence. (Carry on programming as in program sequence — Chapter 5.1).

Stop — The stop command is added to the end of the sequence (**SP** is displayed).

Stop Sequence NP Enter — the new sequence is stored as sequence **NP**.

During program sequence:-

Start — plays the sequence so far.

Play Pattern NN — plays pattern **NN** + returns to previous point in program schedule.

SONGS

PROGRAM SONG

A song is a series of sequences played consecutively. There is only one programming mode — you enter the sequences in the order you want to play them.

Press **Program Song**

Press **Play Sequence 1** **Sequence 2** **Sequence 88** etc

To store the song

Press **Store Song** **Enter**

The song consisting of sequence 1, 4, 88 is stores as **Song 1**.

As with patterns and sequences, you can store 99 songs.

PLAY SONG

To play a song

Press **Play Song** **Enter Start**

The song is played at the various tempo's stored with the individual sequences.

To override these tempos

Press **Play Song Tempo** **Enter**

The song 1 will be played at the tempo set on the front panel tempo controls.

HUMANIZER

On the rear of the SDS 6 is the Humanizer switch and control. This adds a completely new dimension to the feel produced by a machine such as the SDS 6. It introduces playing errors that would occur naturally during any drum performance.

For example, it is rare that a drummer would be able to 'hit' two tom toms exactly together consistently, indeed it can be very effective to delay certain 'hits' to produce a 'flam' effect.

The SDS 6 incorporates 2 delay circuits, which affect Channel 2 and 4 (the shortest delay) and Channel 5 and 7 (the longest delay).

The overall delay of these four channels are controlled by the Humanizer amount knob and the circuits are switched in when the Humanizer Yes/No switch is in the 'Yes' position.

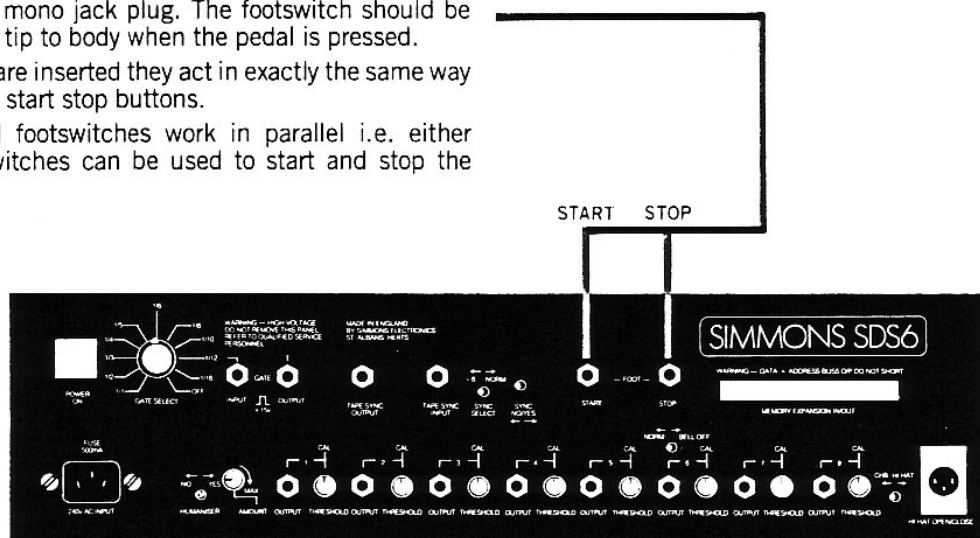
With the switch in the 'Yes' position and the amount knob fully anti-clockwise, only a small amount of delay is discernible. As the amount control is turned further and further clockwise, more delay is introduced in the Channels 2, 4, 5 and 7 and this delay will vary (for a human feel) every time those channels 'hit' the SDS 5 drums. At its extreme clockwise position the dynamics of 'hits' will also vary and the sequencer will even leave out various 'hits' which sounds like many drummers I have played with!

FOOT SWITCHES

The foot switch sockets on the SDS 6 backpanel accept a standard ¼ inch mono jack plug. The footswitch should be wired to short the tip to body when the pedal is pressed.

When the pedals are inserted they act in exactly the same way as the front panel start stop buttons.

The buttons and footswitches work in parallel i.e. either buttons or footswitches can be used to start and stop the sequencer.



The only difference to bear in mind when using the footswitches is that the loop command has to be used before selecting the sequence number when playing a looped sequence.

i.e. normal start is as follows:-

Play Sequence 1 Loop (Chapter 5.2).

As there is no loop footswitch the sequence is played as follows.

Play Sequence Loop 1 Start (Footswitch)

Important

The body of the footswitches is not ground (earth, 0v) do not use as a signal ground. You will disable the front panel switches.

GATES IN AND OUT

IN

The gate input socket accepts positive-going pulses, and will step the SDS 6. 1 position for every pulse.

Thus you can sync the SDS 6 to various sequencers and computers that can supply a pulse.

DIVIDE BY 6

The divide by six function is included so that the SDS 6 can be stepped by sequencers giving a 48 step per bar signal.

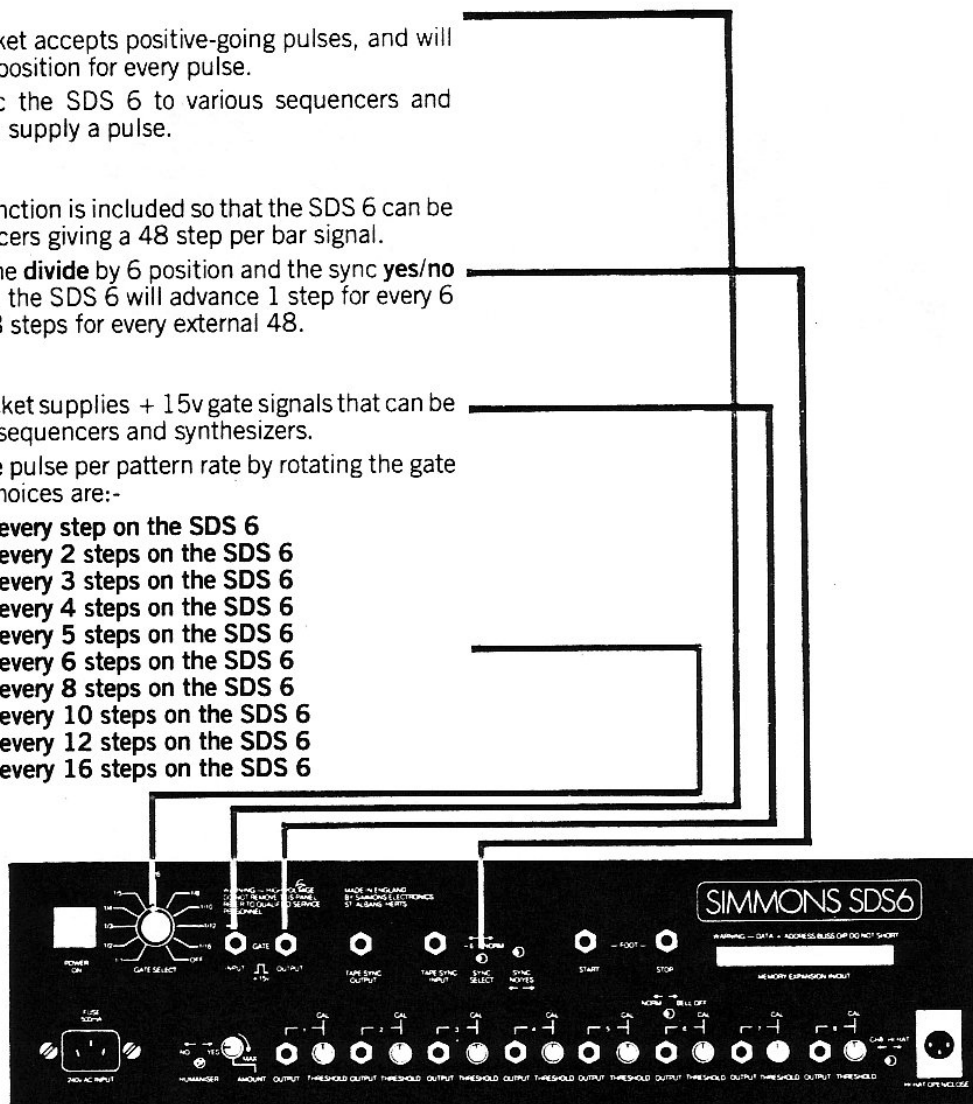
If the switch is in the **divide** by 6 position and the sync **yes/no** switch is set to **yes**, the SDS 6 will advance 1 step for every 6 pulses it sees i.e. 8 steps for every external 48.

OUT

The gate output socket supplies + 15v gate signals that can be used to step other sequencers and synthesizers.

You can choose the pulse per pattern rate by rotating the gate select knob. The choices are:-

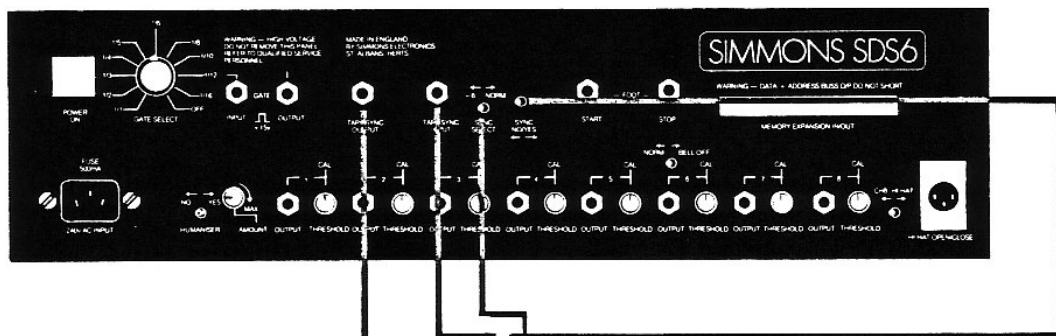
- 1 pulse output for every step on the SDS 6
- 1 pulse output for every 2 steps on the SDS 6
- 1 pulse output for every 3 steps on the SDS 6
- 1 pulse output for every 4 steps on the SDS 6
- 1 pulse output for every 5 steps on the SDS 6
- 1 pulse output for every 6 steps on the SDS 6
- 1 pulse output for every 8 steps on the SDS 6
- 1 pulse output for every 10 steps on the SDS 6
- 1 pulse output for every 12 steps on the SDS 6
- 1 pulse output for every 16 steps on the SDS 6



NOTE: the counters are re-set during the **abort** stage so, if you are repeating a song, sequence or pattern and stepping other sequencers at the same time, **press abort** before you start, this

TAPE SYNC

The SDS 6 provides a coded sync pulse that can be used to sync the sequencer to a previously recorded track.



The tape sync output should be recorded on a separate track along with the drum sounds. Then during playback, if this signal is fed off tape into the tape sync input, the sequencer will play in time with what was previously recorded.

The procedure is as follows — if recording a 'song'.

Recording

Use as many tracks as required to record the drum sounds.

Feed the tape sync output onto a separate track (be careful of cross-talk etc., between the sync pulse and the recorded drum sounds).

Run the tape recorder to obtain a 'leader' from the sync output.

Press Play Song **STOP** Start

When you **press start** the coded sync signal will appear on the sync output.

Playback

Take the sync pulse from tape and feed into the tape sync input socket on the back of the SDS 6.

Switch the sync **yes/no** switch to **yes**

Switch the normal **divide 6** switch to **normal**

Press Play Song **STOP**

Before you **press start**. Start the tape recorder and wait until the sync lead-in signal is present (this ensures that switching transients and start up noise off tape will not introduce a false sync signal) and then

Press Start

The sequencer will then wait until the coded part of the sync signal arrives and then start to step through the song, in synchronisation with the original track.

Uses

This multi-tracking facility allows you to record different drum sounds to be set on the SDS 5, which will then be played in sync with the original drum sounds.

You can of course alter the patterns that made up the original song, creating new drum fills and rhythms that will be played in sync with the original song. e.g. **song 1** may consist of **Pattern 1, 2, 3**, in 4/4 time. If you record this song, change the 'hit information in Patterns 1, 2, 3 to 3/4 time then the two songs will be played in sync forming a complex poly-rhythm (if you change the length of the patterns using **loop**, (See Chapter 7.2) even more complex rhythms can be built up.

Tape Sync Out Level = 500mv p-p

Tape Sync In Sensitivity = 50mv p-p

QUICK PROGRAMMING LIST

Program Pattern

Allows 'hits' to be entered in Matrix. To make new pattern — Store as a pattern number NN (Max 99).

Program Pattern NN

Retrieves pattern NN for alteration. Modified pattern can be stored as new pattern number if required.

Program Pattern Loop

Allows a 'line' to be written into the Matrix to shorten the Matrix length. (Leaves scan speed alone).

Program Pattern Ext. Short Pattern

Same as Loop (above) but slows clock speed.

Program Pattern Start

Allow 'hits' entered in Matrix to be rotated to match start of rhythm with physical start of Matrix.

Program Pattern Tempo

Doubles the speed of the scan line for that pattern.

Program Pattern Set Dynam(ics)

Allows dynamic values 1-9 to be entered against 'hits' in Matrix.

Program Pattern Clear (Clear Clear)

1st clear clears loop line, 2nd clears dynamic values, 3rd clears 'hits' in Matrix.

PROGRAM SEQUENCES—SONGS

Program Sequence. Play Pattern NN Pattern NM Pattern NO

Writes sequence of patterns — NN, NM, NO can be stored as sequence No. NN by store sequence NN.

Program Sequence NN Enter Tempo

Retrieves sequence NN for editing. Enters new tempo (set on tempo controls) value.

Pattern NP, Pattern NQ

Adds pattern NP, NQ to end of sequence NN.

Clears Clear

Clears patterns from end of sequence NN.

Program Sequence Play Pattern NN Enter 99 Times

Creates sequence consisting of pattern NN 99 times.

Program Sequence Play Pattern NN Then Pattern NM NO NP Enter 40 Times

Creates sequence consisting of pattern NN then patterns (NM, NO, NP) 40 times.

Program Song Play Sequence NN, Sequence NM, Sequence NO

Creates song consisting of sequence NN, NM, NO. Can be stored —

Press **Store Sōng NN Enter**

PLAYING BACK PATTERNS

Play Pattern NN Start

Plays Pattern NN once.

Play Pattern NN Loop Start

Plays pattern NN continuously.

Play Pattern NN Loop Start No Start NP Start

Plays pattern NN followed by NO, NP (useful for trying out sequences of patterns 'live').

PLAYING BACK SEQUENCES

Play Sequence NN

Play Sequence Loop NN Start

Plays sequence NN continuously. Use this to loop sequence when using footswitch to start.

Play Sequence NN Loop

As above (Normal looped start).

Play Sequence Tempo NN Start (or Loop)

Overrides programmed tempo with front panel controls.

PLAYING BACK SONGS

Play Song NN Enter Start

Plays song NN.

Play Song Tempo NN Enter Start

Plays song NN but overrides programmed tempo value with front panel tempo controls.

SHOW

Show

Shows memory used so far.

Show Pattern

Shows pattern numbers used so far.

Show Pattern NN Enter

Displays pattern NN in the Matrix (press start to play).

Show Sequence

Shows sequence numbers used so far.

Show Sequence NN Enter

Shows tempo value of sequence NN.

Press **Then** repeatedly to step through sequence. Subsequent presses display next pattern.

Show Song

Shows song numbers used so far.

Show Song NN Enter

Pressing **Then** steps through sequences in the song, displaying the sequence numbers in the keypad.

CLEAR & MISCELLANEOUS

Clear Pattern NN

Clears Pattern NN from memory.

Clear Sequence NN

Clears sequence NN from memory.

Clear Song NN

Clears song NN from memory.

Clear Pattern, Sequence, Song

Clears entire memory.

Clear Clear Pattern Sequence Song

Clears entire memory and computer memory — For use in emergency — See Troubleshooting.

Abort

Stops current function and sets sequencer to switch on state — Resets sync and gate output counters.

Dump Pattern Sequence Song

Dumps entire memory to expansion pack.

Load Pattern Sequence Song

Loads memory from expansion pack (overwrites original memory).

Tempo

Displays readout of tempo value.

TROUBLE SHOOTING

ERROR MESSAGES

There are various error messages that can be generated by the SDS 6 sequencer.

The action that you can take in the event of one of these appearing will vary according to your skill in electronics. They are included here along with their significance for your information however you are advised to contact Simmons Electronics if they do crop up.

CORRUPT

Nothing to do with the Metropolitan Police this, but an indication that something has gone amiss with the organisation of the SDS 6 memory.

It will make itself known to you by flashing all the led's on the front panel every time you try to make a keyboard entry.

The warning is to inform you that the memory has been corrupted and that the computer is in a state of confusion. It cannot guarantee that it will work in a logical fashion or that it will be able to retrieve patterns, sequences, and songs that have been programmed.

This can happen if there has been interruptions of power supply during critical functions, if the internal batteries are flat, if there is an internal electronic failure or if the internal works have been tampered with.

Although a lot of lights flash you will be allowed to continue what you were doing in case you can retrieve the situation, but the computer still knows that it is working with a corrupt memory and the only way to sort things out is to follow the actions as described in the next paragraph, although this means all of your programs will be lost. Try programming and storing a pattern as pattern 1, this can sometimes free the other patterns for use, and at this stage it would be advisable to dump the programs into the SDS 6 memory pack (if the computer will let you).

To clear the memory and reorganise the computer...

Press C Bar C Bar Pattern Sequence S Bar g

The numbers 1, 2, 3, will appear as you press the buttons.

The buttons have to be pressed in that order and the memory is completely cleared and you will have to start programming from scratch.

E0

Unimplemented function. You have somehow managed to enter a routine that is reserved for future expansion. Give us a ring and let us know how you did it.

E1

Internal computer stack full.

E3

Sequence or song too long (maximum is 250).

E4

Internal error.

E5

Too many dynamics see Ch 7.6

E6

Empty sequence is illegal.

E7

Not allowed pattern, sequence or song 0.

E8

Internal timing error.

If any of the above internal error messages crop up we would appreciate a call informing us of the circumstances of its appearance.

WHAT THE HELL WAS THAT—TYPE MESSAGES

SP

Stop message — A stop command has been entered into a sequence — everything is OK. See (Chapter 9.3).

Slight flickering of lights

Normal — It gets worse the harder the sequencer has to work i.e. if there are a lot of dynamics and 'hits' in a pattern. Especially if the pattern is **Ext Short Pattern** at maximum speed.

Tempo slowing down at fast extreme i.e.

x2 tempo — normal the fastest speeds at x2 are outside the capabilities of the SDS 6 design (and are mainly useless) but have been included in case you can use them.

'Hits' or Dynamics missing at fast speeds

Normal — See above.

Drums sound quiet or unbalanced

Check SDS 6 output level controls and adjust to re-balance.

TECHNICAL & DIMENSIONS

Technical

Power consumption 25va (running)

Voltage 220-240v / 110-120v (internally re-wired not customer adjustable)

Current 100 ma @ 240v / 200ma @ 120v

Fuse 500 ma @ 240v la @ 120v

Max Dimensions including feet and knobs
(15.5 x 6.5 x 19.5 inches)

Packing Dimensions

10.75 x 18.5 x 22.75 inches

Unit Weight

9 Kg

Packed Weight

9.5 Kg

Gate Out Signal

+ 14v

Trigger Out Signals

+ .5v — 14v-(Programmable and adjustable in 9 steps)

Tape Sync Output

500 mv P — P

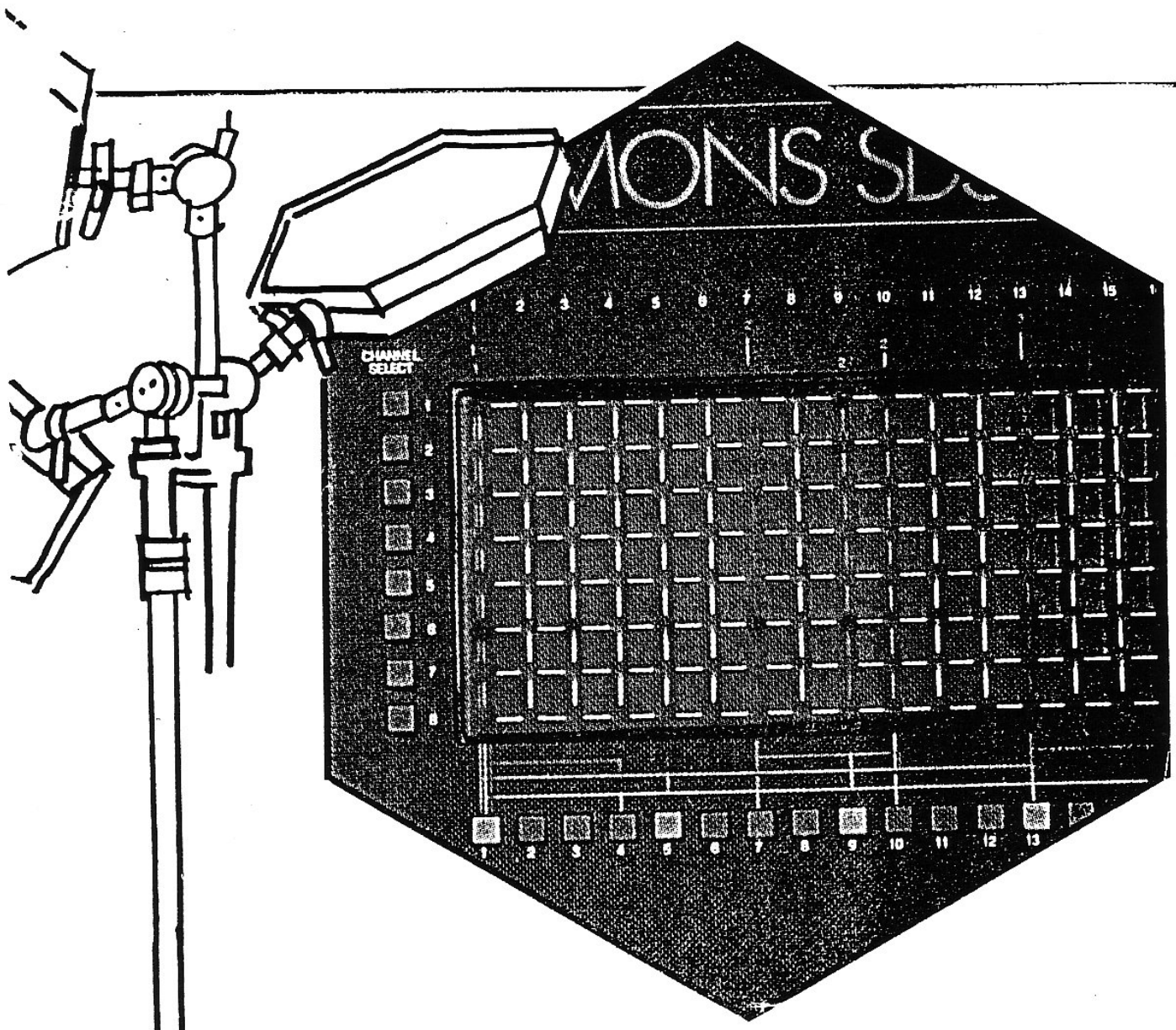
Tape Sync Input Sensitivity

50 mv P - P

SIMMONS SDS6

Designed in England by Simmons Electronics
and Simon Davidmann, without whom, nothing is possible.
Manufactured in England by
Simmons Electronics Ltd.,
Abbey Mill, Abbey Mill Lane, St. Albans,
Herts. AL3 4HG
Tel: 0727 54601/2
Telex: 8952387 Answerback G/Simmons

SIMMONS



SDS6

MIDI GUIDE

SDS6 MIDI GUIDE

This SDS6 MIDI Manual is a supplement to the SDS6 User Manual. It describes the various aspects of using the implementation of MIDI on the SIMMONS SDS6 percussion sequencer.

There are several main sections:-

- o Using SDS6 MIDI.
- o Connecting up.
- o MIDI data transmitted and received.
- o Experimenting with MIDI.
- o Conclusions.

Simmons Electronics Ltd
Unit 11
Alban Park
Hatfield Road
ST ALBANS
Herts
AL4 0JH

Telephone : 0727-36191 (5 lines)

Telex : 291326 HEXDRM G

22nd July 1984

The SDS6 MIDI Implementation

Using MIDI

On the back of the SDS6 are two 5 pin DIN sockets labelled MIDI IN and MIDI OUT. Depending on whether the SDS6 is a master or a slave, either or both sockets can be used. The choice between these is made by the position of the SYNC YES/NO switch on the back panel.

As a master, the SDS6 controls other MIDI instruments. As a slave, it is controlled by other instruments. This MIDI implementation is primarily for syncing purposes between sequencers.

As other MIDI instruments recognize different amounts of MIDI information, the SDS6 can be selected to be in one of several modes for being a master or a slave.

It is important to note that when using MIDI it is not possible to use the SDS6 short pattern extended facility with predictable results.

The Different SDS6 modes

There are 3 MIDI modes: mode 0, mode 1, and mode 6.

In mode 0 and mode 6, stop, start etc are recognized and transmitted - for use with those instruments that have a relatively full MIDI implementation. In mode 1 only timing information is sent and recognized.

In modes 0, 1 four MIDI timing clocks are transmitted and recognized for every vertical note position.

In mode 6, six MIDI timing clocks are transmitted and recognised for every vertical note position. This mode 6 is therefore very useful when requiring to interface the SDS6 to other manufacturers percussion sequencers (drum machines) that use 16 note positions per measure, eg Roland, Sequential Circuits etc.

All three of these can be used in either master (controlling) or slave (being controlled) modes.

Changing SDS6 MIDI modes

To change the mode to mode 0:
press ABORT ENTER 0 ABORT

To change the mode to mode 1:
press ABORT ENTER 1 ABORT

To change the mode to mode 6:
press ABORT ENTER 6 ABORT

The current mode is remembered when the sequencer is switched off, and so it is not necessary to keep resetting the modes.

This MIDI mode setting will be changed when you load from external RAM pack, so remember to re-set it after a load operation.

To use the SDS6 short pattern extended facilities you must be in mode 0 (the default SDS6 mode) and cannot use them with MIDI.

SDS6 Controlling other MIDI instruments - MASTER

- o connect SDS6 MIDI OUT to MIDI IN of other instrument (DIN to DIN lead)
- o set SYNC YES/NO to NO (on back panel)
- o select required mode (eg mode 6)
- o set other instrument to be controlled by SDS6 through MIDI

This allows the SDS6 internal timing circuitry to be used for both the SDS6 and the other instrument.

The SDS6 can now be used as normal. Just before you actually start the SDS6 (eg before pushing START) set up the required pattern on the other instrument. Both instruments should now play together.

Note: When in the master modes you can still use the gate and sync outputs on the back panel.

SDS6 being controlled by other MIDI instruments - SLAVE

- o connect MIDI OUT of controlling instrument to MIDI IN of SDS6
- o set SYNC YES/NO to YES (switch on back panel)
- o set SYNC SELECT to NORM (ie not divide by 6)
- o disconnect any leads plugged into
 - gate input
 - tape sync input
- o set up other instrument to send MIDI information

In this slave mode the internal timing clocks of the SDS6 are disabled - the SDS6 being stepped along by input from the other information through the MIDI IN socket.

The SDS6 can now be used as normal, but after you press start on the SDS6, it will wait for the other instrument to actually start it.

MIDI Data Transmitted and Received

MASTER Received data - all modes (0, 1 and 6)

The SDS6 will ignore any information from the MIDI IN socket while in the master mode (controlling other instruments).

MASTER Transmitted data - modes 0, 6

The following status bytes are transmitted:

TIMING CLOCK (F8H)

- mode 0: 4 are transmitted for every vertical note position. If you require 96 per bar, program the pattern loop position to 25.
- mode 6: 6 are transmitted for every vertical position.

START (FAH)

transmitted when the SDS6 starts playing (eg start pressed), just before the first timing clock.

STOP (FCH)

transmitted whenever the SDS6 stops playing. This is not sent when the SDS6 is aborted. Note that this status is not transmitted when the stop button is pushed, but when the SDS6 stops (ie at the end of the current pattern).

SYSTEM RESET (FFH)

this is transmitted whenever abort is pressed. It is not sent when power is switched on.

MASTER Transmitted data - mode 1

The following status bytes are transmitted:

TIMING CLOCK (F8H)

as for mode 0, but these are only sent when a pattern is actually being played. When the SDS6 stops playing, the timing clocks stop and only continue being sent when the SDS6 starts playing a pattern again.

SYSTEM RESET (FFH)

as for mode 0.

SLAVE Received data - modes 0, 6

The following received status bytes are recognized:

TIMING CLOCK (F8H)

mode 0: the SDS6 will step along 1 vertical note position for every 4 timing clocks received.

mode 6: the SDS6 will step along 1 vertical note position for every 6 timing clocks received.

CONTINUE (FBH)

START (FAH)

these inform the sequencer to actually start playing. No distinction is made between them. After pressing start, the SDS6 waits for one of these before it responds to incoming timing clocks.

STOP (FCH)

when received will signal the SDS6 to stop immediately.

SYSTEM RESET (FFH)

causes the SDS6 to return to the power on condition, as if abort was pushed.

All other received data is ignored.

SLAVE Received data - mode 1

The following received status bytes are recognized:

TIMING CLOCK (F8H)

as for mode 0, but in this mode there is no need to wait for MIDI START or CONTINUE. The SDS6 will start playing after pressing start when the first timing clock is received. It will keep playing as long as timing clocks are received.

SYSTEM RESET (FFH)

as for mode 0.

SLAVE Transmitted data - all modes (0, 1 and 6)

In the slave mode, the SDS6 will transmit any received data through the MIDI OUT socket. (ie. it is also a MIDI THRU socket).

Experimenting with MIDI

While developing the SDS6 MIDI interface we got hold of several other manufacturers equipment to see how well MIDI would actually work. This section explains what was tried, and what the outcome was.

The equipment used:

SIMMONS SDS6	- percussion sequencer
SIMMONS SDS7	- digital/analogue drum synthesizer
ROLAND MSQ700	- MIDI recorder/sequencer
ROLAND TR909	- drum computer
ROLAND JUNO106	- polyphonic keyboard synthesizer
SEQUENTIAL CIRCUITS DRUMTRAKS	- drum computer
SEQUENTIAL CIRCUITS 6TRAK	- polyphonic keyboard synthesizer

Recording - MSQ700, JUNO106, SDS6/7

We started with the SDS6/7 and programmed a sequence of basic patterns (bass, snare and tom fills) with the SDS6 running with its own internal clock. (Though we could have set it to sync from a dummy track being played on the MSQ700 while we were programming the patterns).

We then connected up:

MSQ700 -> JUNO106 -> SDS6/7

<-

The MSQ700 was being the master, and was running with its internal clock. Its MIDI OUT was connected to JUNO106 MIDI IN. The JUNO106 MIDI THRU was connected to the SDS6 MIDI IN. The SDS6 was set to SYNC YES and its internal mode was set to 6. The JUNO106 MIDI OUT was connected back to the MSQ700 MIDI IN (so that the played notes would be recorded).

It is necessary for recording on the MSQ700 for it to be the master as with these instruments it is not possible to be able to get both the note information (from the JUNO106) and the MIDI timing control (from the SDS6) into the one MIDI IN socket on the MSQ700. This is due to the JUNO106 - it has a MIDI OUT and a MIDI THRU. The MIDI OUT just sends the notes etc played on its keyboard, while the MIDI THRU just sends the information that it receives from MIDI IN. If the MIDI OUT could be programmed to also send what comes down the MIDI IN, then we could have used the SDS6 to control the recording via the JUNO106. Other keyboards may allow this.

The MSQ700 was set to record (load) in real time, and the SDS6 was set to play the sequence (start being pressed, but the SDS6 waiting for MIDI start and timing pulses from the MSQ700 through the JUN0106). When the MSQ700 was started it waited (and so did the SDS6) until the first note was played before it started. The metronome on the MSQ700 was giving the timing, and the MSQ700's tempo was then adjusted. When the first note was played the SDS6/7 started up and the MSQ700 started to record. When we stopped the MSQ700 recording the SDS6 stopped.

Playback - MSQ700, JUN0106, SDS6/7

We then set the MSQ700 to playback the recorded piece and started it (without touching the SDS6) - the SDS6 started where it left off, and was in sync but off beat with the played back notes. This is because the MSQ700 was stopped during recording before the end of the SDS6 sequence - the SDS6 always stops immediately MIDI IN tells it to.

If we had stopped recording at the right place and told the SDS6 to play the sequence continually then this would have worked OK. In fact what we should have done, is, before starting the MSQ700 to playback, set the SDS6 to play the sequence from the top (ie abort play seq 4 start - it then waits for MIDI START) and then start the MSQ700. When we did this correctly, the MSQ700, JUN0106 and SDS6/7 played back perfectly.

We must therefore remember to set the SDS6 to start from the beginning of the required pattern, sequence or song every time we start it playing while it is a slave (being controlled by another instrument). If we have stopped the MSQ700 in the middle of playing the piece, and instead of starting the MSQ700 at the beginning with the play button, but use its continue, then both the MSQ700 and the SDS6 continue from where they left off perfectly (ie for continuation of a piece the SDS6 does not need to be touched at all - everything goes down MIDI).

As the SDS6 MIDI OUT socket doubles up as a MIDI THRU socket when the SDS6 is in slave mode, we could have played back with the instruments connected:

MSQ700 -> SDS6 -> JUN0106

Ie, MSQ700 MIDI OUT to SDS6 MIDI IN. SDS6 MIDI OUT to JUN0106 MIDI IN. We then select the sequence on SDS6 and press start, and then start the MSQ700. This configuration still requires us to set the SDS6 to start from the top every time we want to restart, before we start the MSQ700. (We can use continue though if we stopped in the middle).

We could also connect them up with the SDS6 being the master, ie:

SDS6 -> MSQ700 -> JUN0106

We need to set the SDS6 to be a master (back panel switch to SYNC NO), and the MSQ700 to be controlled by MIDI IN.

When we do this and start the SDS6, the MSQ700 starts OK. When we push stop, or abort on the SDS6, the MSQ700 still continues. The problem is that the MSQ700 does not recognise MIDI STOP or MIDI ABORT. We have to stop the MSQ700 manually after stopping the SDS6.

DRUMTRAKS, 6TRAK and SDS6

We tried to use the SDS6 to control the 6TRAK while we recorded a sequence on it but it appears that the 6TRAK requires some MIDI SYSTEM EXCLUSIVE information to actually start recording. It is possible to have the SDS6 controlling the 6TRAK while it plays a recorded sequence.

The DRUMTRAKS could start the 6TRAK recording (as would be expected because they are both from the same manufacturer).

Three Drum Machines - SDS6/7, DRUMTRAKS, TR909

Recording

What we wanted to do was to program some patterns on one machine using its internal clock, and then while that was playing, program a pattern on another while it was synced to the playing one, and also, if possible, be able to program two machines at the same time.

Note that when programming the SDS6 is mentioned it means in the program pattern mode, and not just inserting notes into the pattern display in the abort condition.

We tried various combinations:

- DRUMTRAKS playing -> SDS6 programming
works well but you must get the SDS6 started and waiting for MIDI start before you start the DRUMTRAKS, ie insert the first note, or program an existing pattern.
- TR909 playing -> SDS6 programming
as for the DRUMTRAKS.
- SDS6 playing -> DRUMTRAKS programming
we could not get the DRUMTRAKS to record a pattern when being controlled by MIDI. The problem is that the DRUM TRAKS is not started by MIDI when you are using patterns.
- SDS6 playing -> TR909 programming
works well, but note that pushing abort on the SDS6 will put the TR909 back into power up state (you keep needing to select MIDI sync in).
- SDS6 programming -> DRUMTRAKS playing
works well, but the DRUMTRAKS must be playing a song. It is easy enough to create a one pattern song. Note that the SDS6 must be put into program pattern mode after the DRUMTRAKS song and MIDI sync are selected.
- SDS6 programming -> TR909 playing
works well.
- SDS6 programming -> TR909 programming
works well.

Playing

After programming some patterns we set up some sequences on the machines and tried various combinations of connecting them together:

- SDS6 - TR909 - DRUMTRAKS
this worked well, but the DRUMTRAKS must be set to play a song. When the DRUMTRAKS is playing just single patterns it does not recognize MIDI start. (It was possible to push the start buttons on the SDS6 and DRUMTRAKS together most times, but it is better to have a song on the DRUMTRAKS and just push the SDS6 start button).

SDS6 -> DRUMTRAKS -> TR909
works well (note as above).

DRUMTRAKS -> SDS6 -> TR909
works well, but the DRUMTRAKS appears to send a MIDI CONTINUE command at the beginning of every song. This means that if you have a song on the DRUMTRAKS that contains just one pattern, a continue command is send every pattern. This may cause problems if you stop one of the other machines - they then start again (continue) at the beginning of the next DRUMTRAKS song (as it being played continuously).

TR909 -> SDS6 -> DRUMTRAKS
works well, but when we stopped the TR909 and then used the continue the DRUMTRAKS did not continue.

Connecting them all together

All the instruments were then connected together in a variety of ways. With what was learnt above, eg DRUMTRAKS pattern starting etc, a lot of what happened was predictable. We were playing the notes that had been recorded on the MSQ700 earlier.

MSQ700 -> SDS6 -> 6TRAK -> DRUMTRAKS
the DRUMTRAKS had to be started (by pressing start) at the same time as the MSQ700. The 6TRAK only has MIDI OUT and no MIDI THRU and MIDI START did not appear to be transmitted through it.

MSQ700 -> 6TRAK -> SDS6 -> DRUMTRAKS
not useable as the SDS6 needs MIDI START when syncing, and the 6TRAK does not appear to send one.

MSQ700 -> JUNO106 (OUT) -> SDS6 -> DRUMTRAKS
THE JUNO106 MIDI OUT was connected to the SDS6 MIDI IN.
THE SDS6 and DRUMTRAKS did not work when the MSQ700 was started. This is because the JUNO 106 MIDI OUT does not send what it gets from MIDI IN but just the note information that is played on it.

MSQ700 -> JUNO106 (THRU) -> SDS6 -> DRUMTRAKS
everything worked well. JUNO106 MIDI THRU passing MIDI START etc through to the SDS6 and DRUMTRAKS.

MSQ700 -> JUNO106 (THRU) -> SDS6 -> 6TRAK -> DRUMTRAKS
worked OK if you pushed MSQ700 and DRUMTRAKS start at the same time. (again, the problem appeared to be the 6TRAK not sending out MIDI START).

MSQ700 -> JUNO106 (THRU) -> SDS6 -> 6TRAK -> DRUMTRAKS -> TR909
worked OK, but again anything after the 6TRAK needs manually starting. With the TR909 connected after the DRUMTRAKS, we only needed to start the MSQ 700 and the DRUMTRAKS together because the DRUMTRAKS sent out a MIDI START to the TR909.

MSQ700 -> JUNO106 (THRU) -> SDS6 -> TR909 -> DRUMTRAKS -> 6TRAK
most worked, but the 6TRAK did not receive anything. This being because the DRUMTRAKS and the TR909 do not have MIDI THRU sockets and so kind of absorb note information.

MSQ700 -> JUNO106 (THRU) -> SDS6 -> DRUMTRAKS -> TR909 -> 6TRAK
worked but the 6TRAK was not playing the notes that the MSQ700 was sending. It was in fact playing the external instrument information that had been programmed into the TR909's patterns.

MSQ700 -> JUNO106 (THRU) -> SDS6 -> DRUMTRAKS -> 6TRAK
nothing came out of the 6TRAK at all because of the DRUMTRAKS absorbing note information.

MSQ700 -> SDS6 -> JUN0106 (THRU) -> DRUMTRAKS - 6TRAK
the JUN0106 worked OK through the SDS6 but 6TRAK received nothing
from the DRUMTRAKS.
DRUMTRAKS -> SDS6 -> MSQ700 -> JUN0106 (THRU) - 6TRAK
everything worked OK, but if we stopped the DRUMTRAKS mid-way through
the sequence, then some of the notes on the 6TRAK stayed on.

Conclusions

Basically, we managed to get around most of the shortcomings of the various
MIDI implementations by connecting the instruments up in different ways
(basically, by putting the awkward ones at the end of the chain!)

There are several points worth noting:

DRUMTRAKS:

Must always be playing a song when controlled by other instruments.

SDS6:

must be in MIDI mode 6 for connecting to DRUMTRAKS and TR909.

As a slave it must be aborted and started from the top each time.

The MIDI OUT socket is a MIDI THRU socket when used as a slave.

DRUMTRAKS, TR909, 6TRAK:

These do not have a MIDI THRU socket, and so are difficult to use in
the middle of a chain where note information is being passed.

AND REMEMBER:

Switch back to SDS6 MIDI mode 0 before using short pattern extended,
EVEN THOUGH you are now not going to be using MIDI.

Thanks for visiting
<http://www.simmonsmuseum.com>

