Design Techniques for

Rhythm generators, for use with electronic organs or as an adjunct to a musical group, can add a great deal to the final musical effect. And, as an exercise in electronic design, they present an intriguing challenge with almost unlimited scope for the designer's imagination. In this and a subsequent article the author discusses both the theory and practice of these circuits.

One of the unfortunate aspects of modern technology is that the home constructor who uses large-scale integrated circuits, probably hardly understands what he is doing. Somewhere, a highly specialised designer channelled all his experience into a package, adding instructions as to what to connect to which pin to make "it" work.

There are however more problems than just this one. As the chips become more specialised, it becomes harder for the average hobbyist to buy the particular IC he needs. If he is lucky enough to find what he wants, he is usually faced with yet another disadvantage. The end result of his project is pre-determined by the designer of the chip and, as it is usually "hard-wired", the home constructor has no way of altering the design to suit his particular purpose.

This may be acceptable for "run of the mill" projects like amplifiers and radios where the aspect of personal taste is mostly directed to the cabinet in which the unit is housed; it is an entirely different matter with more creative instruments like organs and rhythm units. The rhythm patterns created by a rhythm chip are seldom to the liking of the user and always monotonous.

What is needed is freedom of choice to enhance the creativity of the hobbyist. Especially when playing an instrument, the need for new and different rhythm patterns occurs all the time. It should be possible to listen to a record, to analyse the rhythm pattern and to simulate this on an electronic rhythm unit. To accommodate this need, the author started to experiment with rhythm generators, setting the following objectives:

- full flexibility in rhythm programming.
- Simple circuitry, using only easy-to-buy components.

Some of the results of these experiments are discussed in this article. The emphasis of this discussion will be on rhythm pattern generation rather than on instrument simulation, although some examples of these will also be given.

The most basic rhythm generator is a clock that produces a constant stream of "ticks". In a somewhat modified version, the clock is called a metronome. Here, by changing the length of the pendulum, we can alter the tempo of the ticks to suit the music played. We can play any piece of music against these ticks as there is nothing that differentiates one tick from the other.

Waltz rhythms

Let us assume that we want to play a waltz. This kind of melody is built upon a rhythm consisting of three distinct "ticks", the first of which (the "down beat") has more emphasis, eg [BOOM – TICK – TICK] [BOOM – TICK – TICK] (etc). A drummer would hit the bass drum on the BOOM and the snare drum on the TICKS. Each hit would be separated by exactly the same time interval to keep the tempo constant. In musical terms, this is called "three beats in the measure", "three beats in the bar", or "3/4 time". The measure (or bar) is a given period of time in which the three beats have to occur. Obviously, each beat in this example of the basic waltz represents 1/3 of a measure (bar). (Fig. 1).

Rather than discussing other and more complex rhythm patterns at this stage, let us stick to this simple example for a while and replace it with an electronic device, performing the same task. Assume we have two electronic circuits capable of producing a sound through an amplifier-loudspeaker system, one like a bass drum, the other like a snare drum. Each time the circuit receives an electrical pulse at its input, the respective sound is produced. In other words, the hit by the drummer is now replaced by the "hit" of the pulse.

To provide the "hits" we require a pulse generator. In its simplest version, this could be a square wave generator

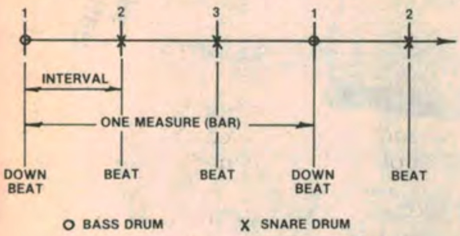


Fig. 1

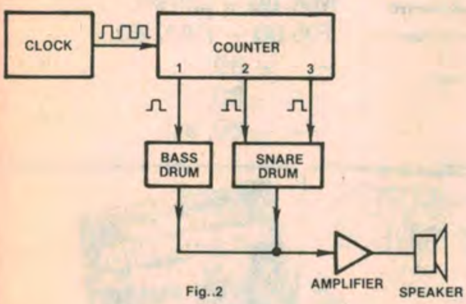


Fig. 2

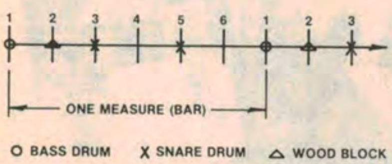


Fig. 3

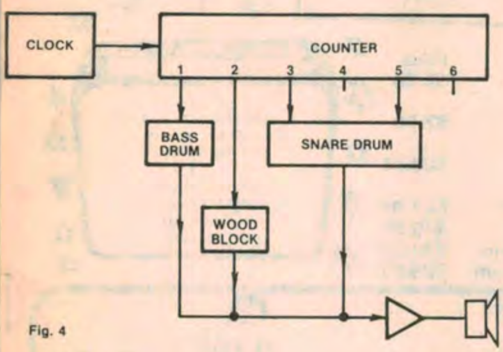


Fig. 4

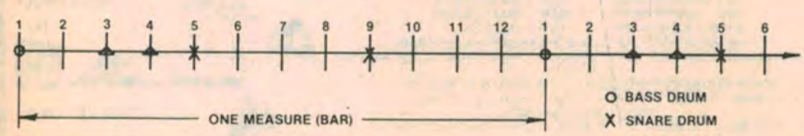


Fig. 5

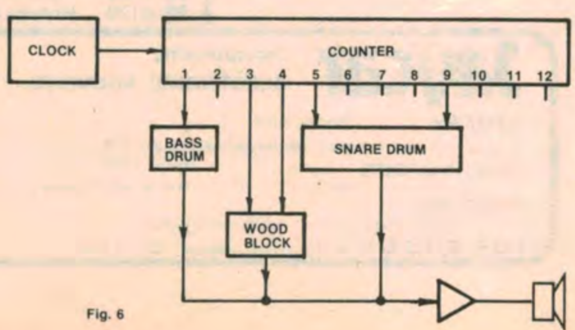


Fig. 6

Rhythm Generators

by NICK LABORDUS*

*5/3 Durham C1, North Ryde, NSW 2113

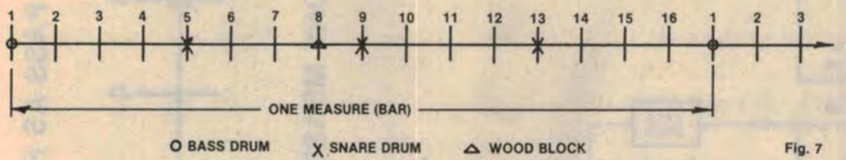


Fig. 7

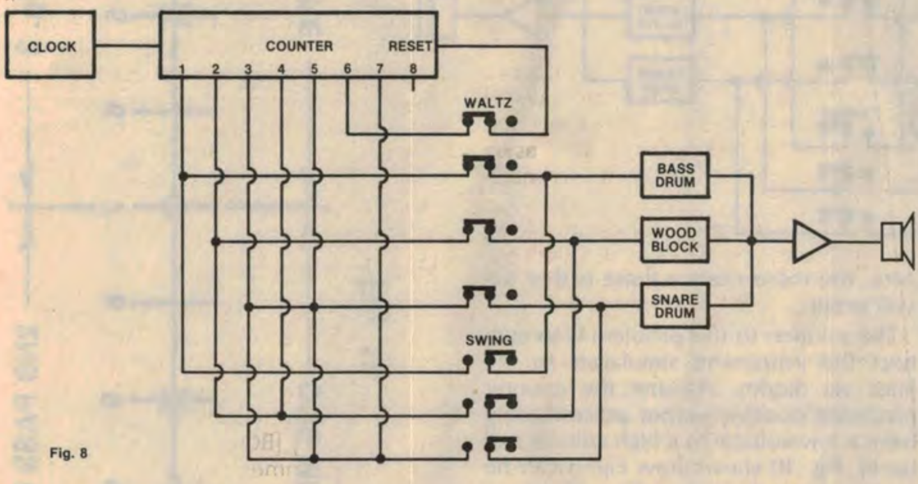


Fig. 8

Fig. 9

built around a 555. This unit produces a constant stream of pulses at its (one) output line. Connecting this output to the input of either sound simulator, we recreate the metronome effect ie, a constant stream of BOOMS or TICKS but nothing like the required BOOM-TICK-TICK.

Sequencing

Obviously, the sound simulators cannot distinguish between any of the incoming pulses. A way to overcome this is to separate the pulses by channelling them sequentially into separate output lines. This can be achieved by feeding the pulses into a counter with (for this example of a waltz) three output lines. The first pulse will activate line 1, the second pulse will activate output 2 and the third one output line 3.

The incoming pulses are now separated into three distinct output lines, each of which will produce a pulse in sequence. If we connect output 1 to the bass drum circuit and outputs 2 and 3 to the snare drum circuit, we do get the required effect: [BOOM – TICK – TICK] [BOOM – TICK – TICK] etc. The block diagram of this "rhythm unit" is given in Fig. 2. The way we connect the counter output lines to the sound simulators determines the rhythm pattern. The frequency of the pulse generator (clock) determines the tempo.

For the moment, let us stay with the waltz, but ask the drummer to give some variations to the old BOOM – TICK – TICK which by now is getting a bit boring. Assume he hits a woodblock just halfway between the bass drum and first snare drum hit. This would sound like: [BOOM – TOCK-TICK – TICK] [BOOM – TOCK-TICK – TICK] etc. The woodblock "TOCK" sound appears in the measure at 1/6 of the time (Fig. 3).

To re-create this sound electronically we have to add a woodblock simulator to our two instruments. It will be clear by now that we also need a different counter, ie one with six output lines to allow us to pick the 1/6 time interval. In this counter the 7th input pulse will appear on line 1 again. Fig. 4 shows the block diagram of this rhythm unit. Note that only four of the six output lines are used in this example and that the clock must run at double the frequency to maintain the same tempo, as the pulses must pass through double the number of counter stages.

Not satisfied with this, our drummer introduces a further variation and gives the woodblock another hit between the first woodblock hit and the first snare drum hit. Without any further explanation it will be clear that this last hit is 1/12 of a measure (Fig. 5). Just for the sake of this one extra hit we have to use a 12-output line counter in our electronic equivalent, most of which will remain unused in our example. The clock has now to run at four times the frequency of the first (three line) example to produce the same tempo. Fig. 6 gives the block diagram of this rhythm unit. Obviously, by having 12 output lines available, we can now program a large number of other variations, like the drummer using his fantasy to create new and exciting effects.

Complex rhythms

So far we have discussed only those rhythm patterns with three basic beats in the measure. A lot of music is based upon a basic two beats in the measure (eg, a march and a two-step) and even more on four beats in the measure (eg, foxtrot, swing, tango, etc). It will not be difficult to understand that for such patterns we need a counter with four, eight, or 16 output lines, following the same examples as above. Fig. 7 shows a 16 line pattern for a swing rhythm sounding like [BOOM – TICK KA-TICK – TICK] [BOOM].

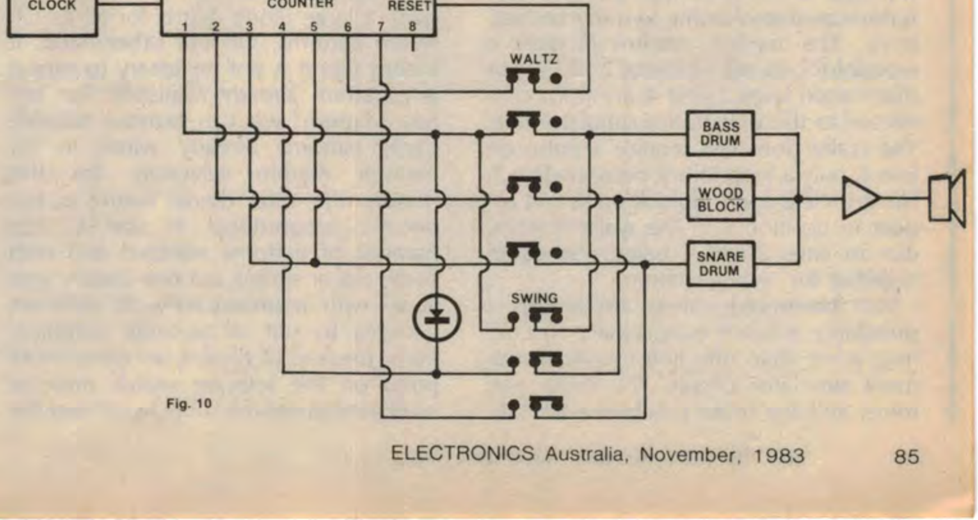


Fig. 10

Design Techniques for Rhythm Generators

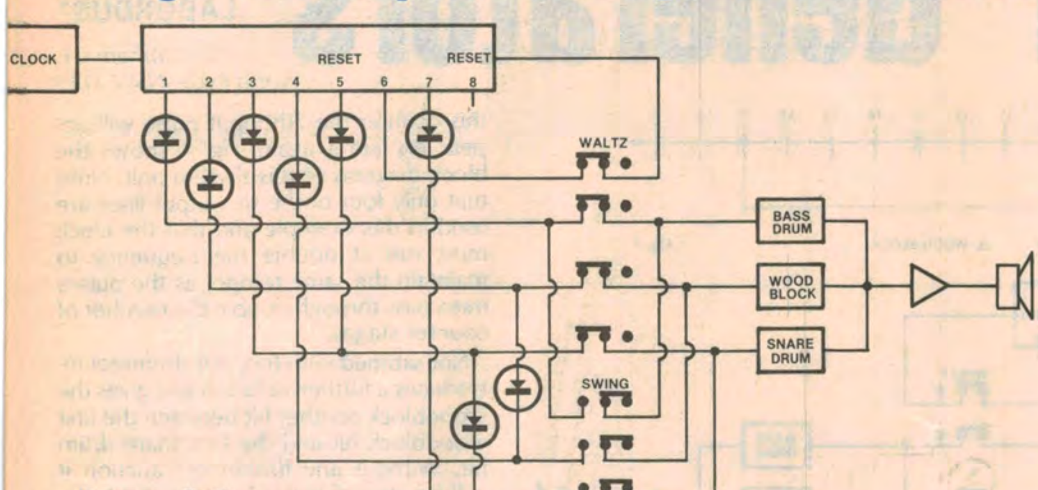


Fig. 11

Most counters can be forced to return to line 1 before they have reached the last counting line. This means that we can use a 16 line counter for a waltz rhythm by "resetting" it after line 12. Keep in mind that this reset pulse must force the counter into its zero position before the next input pulse arrives, otherwise the rhythm unit will produce a hiccup after each measure. This is achieved by using the trailing edge of the clock pulse for the reset function.

Fig. 8 shows a rhythm unit that allows selection between two rhythm patterns, one for the waltz, the other for the swing. The switches used cancel each other, so that only one of the two patterns can reach the instrument simulator circuits at the time. In Fig. 8, the waltz switch is "on" and the swing switch is "off". Note that we have chosen the eight line version for simplicity's sake. The swing pattern is wired according to Fig. 9 and the waltz according to our previous example (Fig. 3). Everything looks fine and one would expect this to work. Regrettably it does not.

Unwanted effects

There are two basic reasons why Fig. 8 will not work. The first concerns interference between the two rhythm patterns. The "swing" pattern dictates a woodblock sound on beats 2 and 4. For this reason lines 2 and 4 are both connected to the woodblock simulator line. The waltz does not require a pulse on line 4, only a woodblock pulse on line 2. Nevertheless a woodblock pulse will appear in position 4 in the waltz rhythm, due to lines 2 and 4 being connected together for "swing" pattern.

Such unwanted effects are always a possibility when it is necessary to connect more than one line to one instrument simulator circuit. The more patterns, and the more complex each pattern, the more chance there is that this will occur.

The solution to this problem is to connect the instrument simulators to the lines via diodes. Assume the counter produces positive output pulses (going from a low voltage to a high voltage and back). Fig. 10 shows how Fig. 8 can be modified to solve the problem involving lines 2 and 4. Similar problems, such as might arise with other configurations, may require the other simulator lines to be similarly isolated.

The second problem concerns the loading effect on the counter outputs where two outputs are joined together, eg, pins 3 and 5 in Fig. 8. Here the problem is that pin 3 cannot go high whilever pin 5 is being held low, and vice versa. The solution to this problem is to isolate each output pin with a diode, the final modified circuit being shown in Fig. 11. This circuit will work. (Although we have separated these two problems, for ease of explanation, both can in fact occur together, as in the case of pins 2 and 4, depending on the various rhythm patterns involved.)

All this means that rhythm pattern programming has to occur via diodes. A large rhythm unit will therefore have quite a large diode matrix for all its different patterns. On the other hand, it means that it is not necessary to repeat any pattern already available. For any new pattern, we can borrow suitable (part) patterns already wired in for another rhythm selection. For this reason, the total diode matrix is not directly proportional in size to the number of patterns selected and with some clever sorting out one usually gets away with approximately 20 different patterns to suit all possible combinations. (Instead of diodes, an extra set of poles on the selector switch may be used to separate the "borrowed" and the

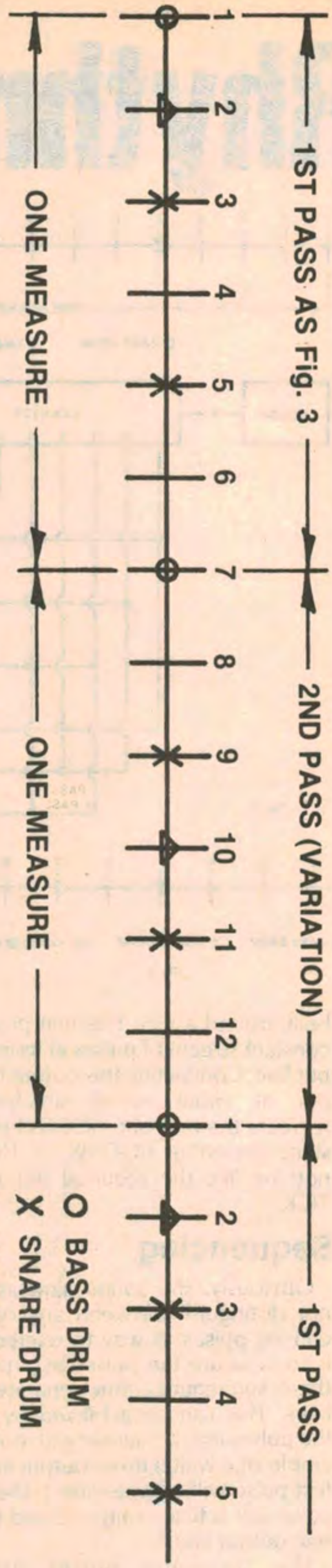


Fig. 12

"additional" pattern.)

The rhythm unit described above can produce a waltz or a swing rhythm in a pre-programmed pattern. It would be possible to re-arrange the diodes in a different manner to create other patterns. For each rhythm added (eg, a foxtrot or a tango), an additional selector switch is required. For any additional instrument

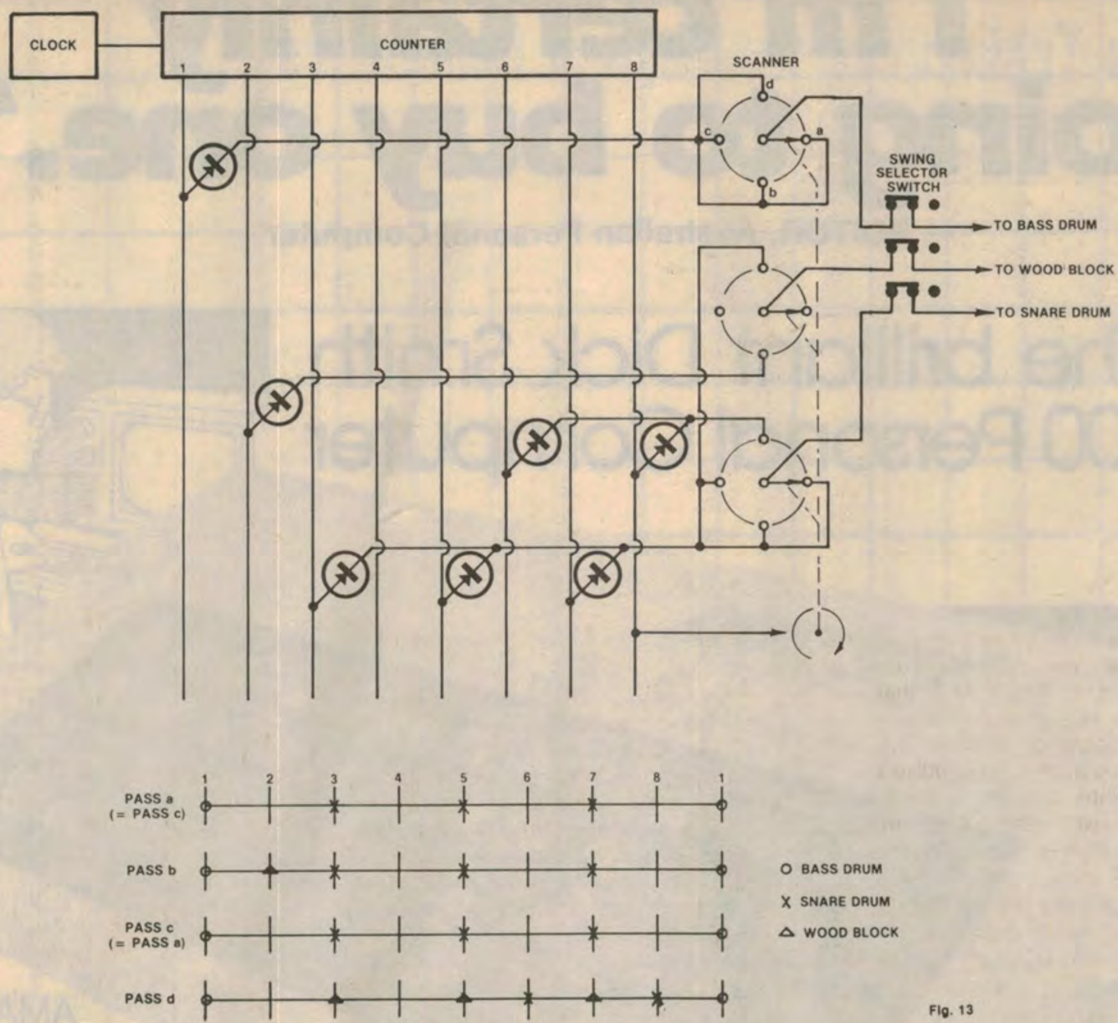


Fig. 13

Design Techniques for Rhythm Generators

simulator, each switch must have one more set of contacts.

This means that a complex rhythm unit may require, say, 10 selector switches, each with six to eight sets of poles to accommodate all available instruments. Here we strike the problem of general availability of such multi-pole switches. The average home constructor is forced to buy from retailers as the quantities required by him are low. The chances of finding a retailer who stocks the exact switch required are virtually zero.

Having encountered this problem, it was decided to steer away from multi-pole switches and use only easy-to-buy single pole versions. These were used to drive (equally easy-to-buy) electronic switches which come in IC packages of four at a time (type 4016). In this way, the number of poles can be extended indefinitely and all problems are solved. The cost is also lower, which is an additional advantage.

All rhythm units described so far have a common disadvantage. Each rhythm pattern selected will be repeated again and again. This monotony can become quite irritating and is the reason why so many rhythm units remain unused after a while.

If we return for a moment to the pattern Fig. 3, using six output lines, we could program that same pattern on the first six lines of a 12 line counter, leaving the second six lines free. We could use these free lines to program a variation on the pattern, as illustrated in Fig. 12. This gives two different "passes" for the same rhythm selection, reducing the monotony a little.

At the same time, however, we have sacrificed the possibility for a 1/12 beat. This could be overcome by using a 24-line counter, programming the first 12 lines for one pattern and the other 12 for a variation. We could add another variation by using a 36-line counter and reduce the monotony further by adding more and more lines. This results in very complicated wiring and a lot of money spent on counters. It is not a practical solution.

Another way to approach this problem is to introduce yet another counter, acting as a scanning switch. (Not to be confused with the rhythm selector switches.) While the rhythm selector switch stays in the chosen position (eg, "tango"), the scanning counter switches from one relevant rhythm pattern to the next variety each time the counter has reached its final line (and before the next input pulse reaches the counter). The number of output lines of the scanner counter determines the maximum number of pattern variations that can be created per chosen rhythm type.

In Fig. 13 the scanner is shown as a constantly rotating multi-pole switch. In practice, the scanner is not a switch (shown for easy understanding) but a counter, the output lines of which sequentially enable certain diode programs to be connected to selected instrument simulators via electronic switches.

In Part 2, some practical circuits will be discussed, ranging from a simple 16-line one pass rhythm generator to a scanning version. Some circuits for instrument simulators will be discussed and a few examples of practical diode matrices will be given to allow the home constructor to start his own experiments.

Design Techniques for

In Part 1 of this article we discussed basic rhythm structures and established the principles of rhythm generation using clock oscillators, counters and instrument simulators. From this theoretical discussion we now move on to practical circuits which can exploit these ideas and serve as a basis for further experimentation.

As we established in part 1, the basis for every rhythm unit is a device that generates pulses sequentially on separate output lines. One can obtain rhythm effects with as few as three or four output lines but to allow for more interesting variations 16 lines must be considered the minimum and for very fancy rhythm patterns one may need 32 lines to allow for 1/32 beats. Rather than just give one circuit, we shall discuss several approaches; a basic 16 line version, a 32 line version, and a scanning multi-pass circuit. We shall also give some diode matrix rhythm pattern examples.

One word of warning. If you decide to start experimenting with pulse generating circuits, make sure you have an oscilloscope. Without this instrument it is almost impossible to track down faults in the circuit if it does not work.

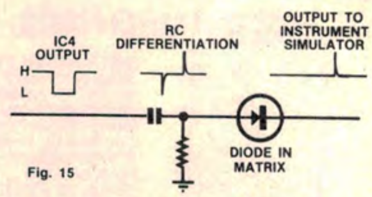
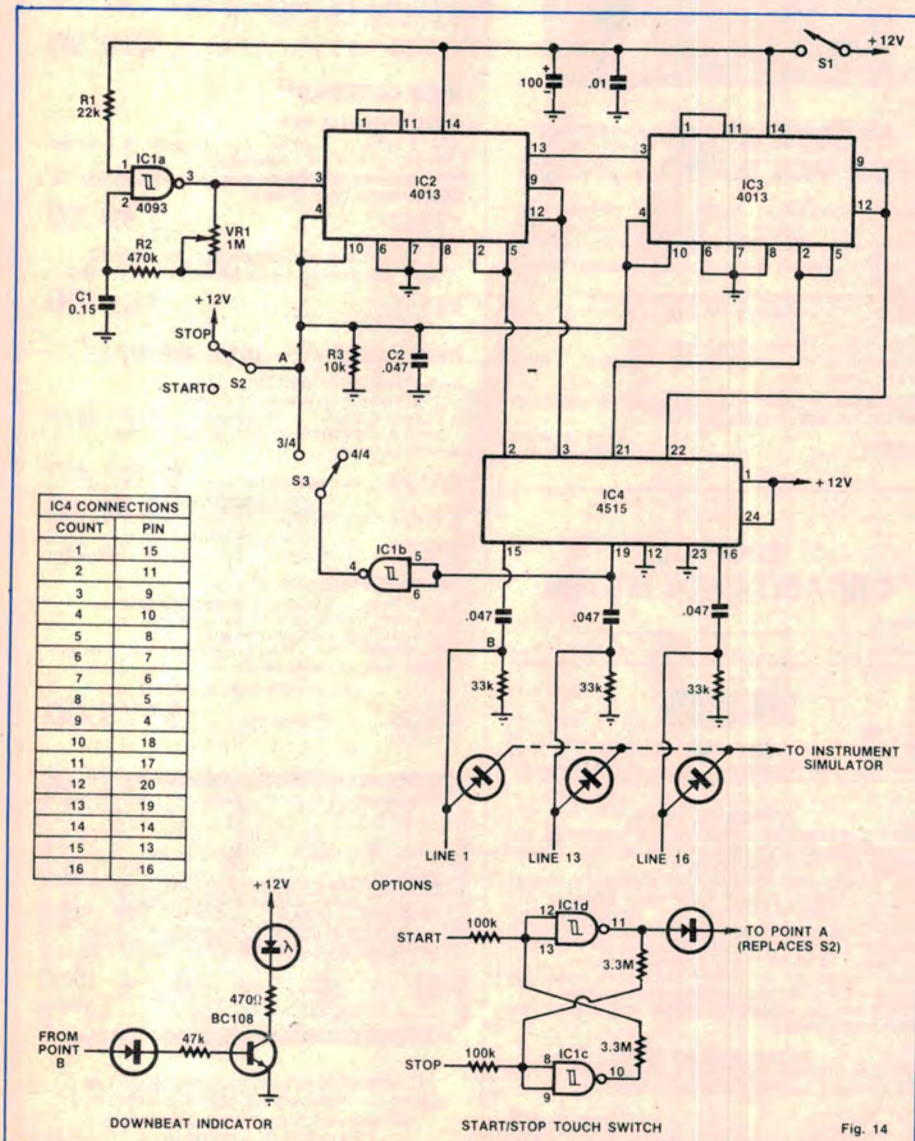


Fig. 15

The complete circuit of a basic single pass 16 beat pulse counter is shown in Fig. 14. The heart of the circuit is the combination of IC2, 3 and 4 which together form a counter with one clock frequency input and 16 output lines. IC2 and 3 are dual D-type flipflops (4013) which together form four flipflop stages in cascade. The Q output of each stage is used to provide the four input lines to IC4.

Depending on the state of each flipflop, the four lines carry a code of high and low voltages. These codes are accepted by IC4, a 4-to-16-line decoder, type 4515. Each input code results in one of the 16 output lines being activated. We shall therefore require a sequentially changing code to obtain the required effect of the 16 output lines to be activated in sequence.

The outputs of IC4 are normally high and will go low one by one when the ap-



IC4 CONNECTIONS	
COUNT	PIN
1	15
2	11
3	9
4	10
5	8
6	7
7	6
8	5
9	4
10	18
11	17
12	20
13	19
14	14
15	13
16	16

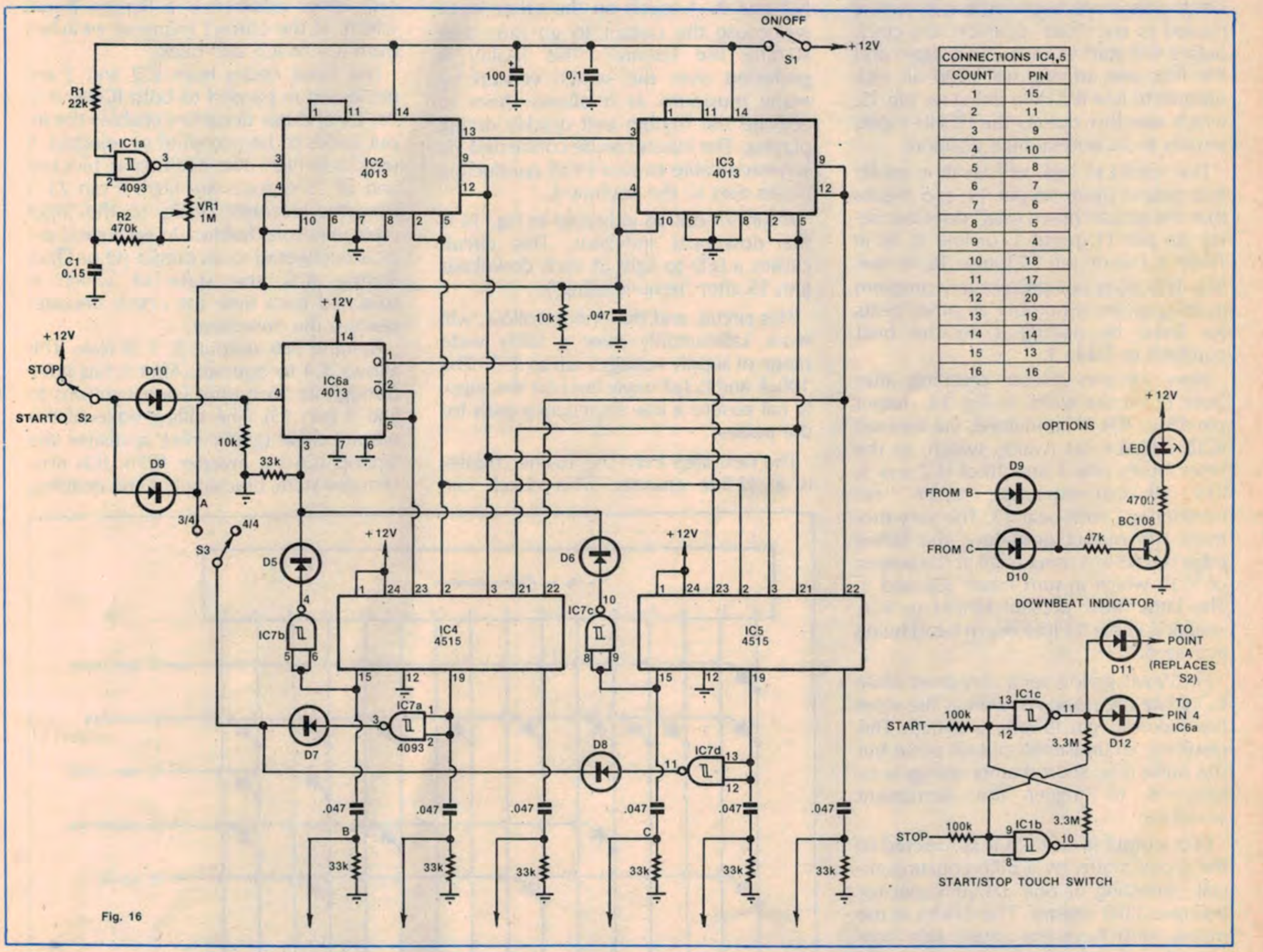
Fig. 14

TABLE 1

Input	Output IC 2 and 3				IC4
pulse number	stage 1	stage 2	stage 3	stage 4	pin low
1	L	L	L	L	11
2	H	L	L	L	9
3	L	H	L	L	10
4	H	H	L	L	8
5	L	L	H	L	7
6	H	L	H	L	6
7	L	H	H	L	5
8	H	H	H	L	4
9	L	L	L	H	18
10	H	L	L	H	17
11	L	H	L	H	20
12	H	H	L	H	19
13	L	L	H	H	14
14	H	L	H	H	13
15	L	H	H	H	16
16	H	H	H	H	15

Rhythm Generators *Part 2*

by **NICK LABORDUS** *5/3 Durham Close, North Ryde NSW 2113*



CONNECTIONS IC4,5	
COUNT	PIN
1	15
2	11
3	9
4	10
5	8
6	7
7	6
8	5
9	4
10	18
11	17
12	20
13	19
14	14
15	13
16	16

Fig. 16

propriate input code is presented. As we shall see later, the instrument simulators require a sharp pulse at their input. This means that the output signals of IC4 must be differentiated (Fig. 15). Differentiation of a square wave by an R/C network results in two pulses, a negative one on the falling edge of the square wave and a positive one on the rising edge. Only one of these two can be used to activate the instrument simulator and a decision must be taken before the counter circuit can be designed.

Either version is possible but as there is a time shift between the positive and the negative pulse, the choice has some effect on certain reset connections in the counter circuit. In our example we have selected instrument simulators with a positive input pulse.

In Fig. 14, IC1a and its associated components form the clock pulse generator, the frequency of which is determined by the values of C1, R2 and VR1. The clock produces square wave pulses which are fed into four flipflop circuits in cascade. Both IC2 and IC3 consist of two flipflops each. Each flipflop stage divides its input pulse frequency by two. At a clock frequency f, the respective output frequencies of the stages 1 to 4 are: Stage 1 (pins 2, 5 IC2) f/2; stage 2 (pins 9, 12 IC2) f/4; stage 3 (pins 2,5 IC3) f/8; stage 4 (pins 9, 12 IC3) f/16.

Table 1 shows the behaviour of the four output lines as new pulses from the clock generator arrive at the input of flipflop 1. Table 1 also shows which of the 16 output lines of IC4 (indicated by pin number) goes from high to low at what input code. Note that the flipflops return automatically to the first code after having reached the 16th. This suits our aim perfectly for all non-waltz rhythm patterns. For waltz pattern we need only 12 lines and we shall have to reset the counter after it has produced the 12th output pulse.

But before considering this function in detail we will have to modify the beat (input pulse) numbers of Table 1. This is due to the need to always start the rhythm generator on beat 1.

Using the "on-off" switch to start the unit will result in the counter starting in a random fashion rather than on beat 1 (the down beat). This can be overcome by introducing a separate switch for the "start-stop" action, leaving the "on-off" switch in the "on" position so that all circuits are functional.

In the "stop" position, S2 connects a high to reset pins 4 and 10 of IC2 and 3. This forces the four selected outputs to show HHHH, independent of the arriving clock pulses. As soon as the switch is moved to the "start" position, the clock pulses will start to take effect again and the first one arriving will send all four outputs to low (LLLL). In doing so, pin 15, which was low due to the HHHH input, returns to its normal high position.

This return to high will result in an active output pulse on pin 15, and means that the actual "down beat" does not occur on pin 11 (pulse 1, or line 1), as in Table 1, but on pin 15 (pulse 16, or line 16). This does not present any problem as long as we re-number all other beats (or lines) by adding 1 to the beat numbers in Table 1.

Now we can discuss resetting after beat 12 for the waltz. In Fig. 14, output pin 19 of IC4 is connected, via inverter IC1b and the 3/4 (waltz) switch, to the reset inputs pins 4 and 10 of IC2 and 3. Pin 19 corresponds, after "re-numbering", with beat 13. The very moment this output goes low, the falling edge results in a rising edge at the output of IC1b which in turn resets IC2 and 3. The latter now present HHHH to IC4, resulting in pin 15 (the down beat) being activated.

Pin 19 will go low for a very short while to initiate the reset. As soon as the reset has occurred, pin 19 returns to high. This results in an unwanted output pulse but this pulse is so short that its energy is insufficient to trigger the instrument simulator.

Each output line of IC4 is connected to the diode matrix by a differentiating circuit consisting of one .047μF capacitor and one 33kΩ resistor. The diodes in the matrix must have the anode side connected to the output lines so that only the positive pulses will arrive at the instrument simulators.

Having two spare gates in IC1, we can add a touch control to our rhythm unit without much effort. Touching one of the inputs with the finger will produce a high output, acting as "stop" by resetting IC2 and 3. A touch on the other input will cause the output to go low, thus starting the counter. This facility is perferred over the switch version by many musicians as it allows them to activate the rhythm unit quickly during playing. The inputs can be connected via screened cable to two small conducting plates next to the keyboard.

A further option indicated in Fig. 14 is the downbeat indicator. This circuit causes a LED to light at each downbeat (pin 15 after "re-numbering").

This circuit, and the one to follow, will work satisfactorily over a fairly wide range of supply voltages, up to 15V. The 100μF and 0.1μF capacitors on the supply rail ensure a low impedance path for the pulses.

The next step from the 16-line counter is a 32-line counter. This circuit can create fancy rhythm patterns with 1/32 beats or act as a double-pass 16-beat counter, allowing some reduction of monotony. The full circuit is shown in Fig. 16. The heart of the circuit is the same as that of the 16 beat counter. In this case there are two 4-to-16-line decoders to obtain 32 output lines. The only other addition is a flipflop (IC6a) which, at the correct moment, switches from IC4 to IC5 and back.

The input codes from IC2 and 3 are presented in parallel to both IC4 and 5. Pin 23 of these decoders enables the input codes to be accepted or rejected. If pin 23 is high, the decoder is blocked and all 16 outputs are high. If pin 23 is low, the decoder reacts to the input code in normal fashion. In each case, pin 23 is connected to an output (Q or $\overline{Q}$) of flipflop IC6, the state of which is switched each time the active decoder reaches the downbeat.

Assume IC6 output 2, 5 is low. This allows IC4 to operate. After it has gone through its 16 output lines it returns to line 1 (pin 15). The falling edge of this output pulse (going low), activates the flipflop IC6 via inverter IC7b. IC6 now changes state, blocking IC4 and enabling

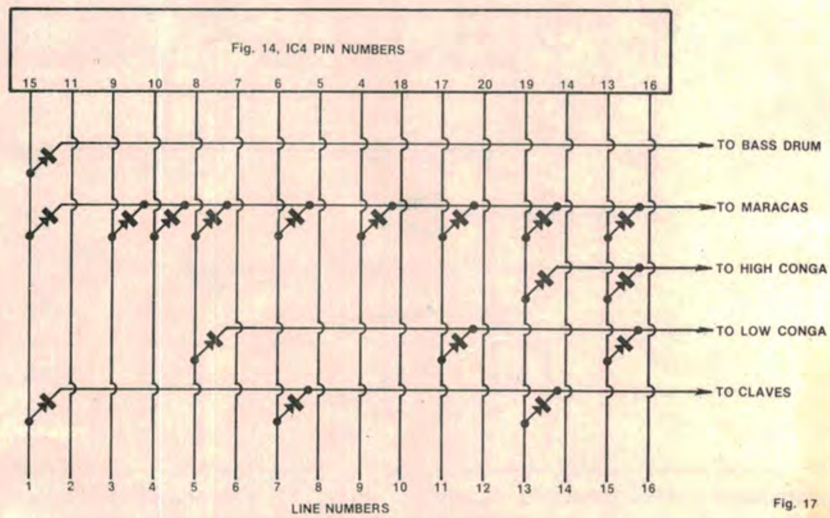
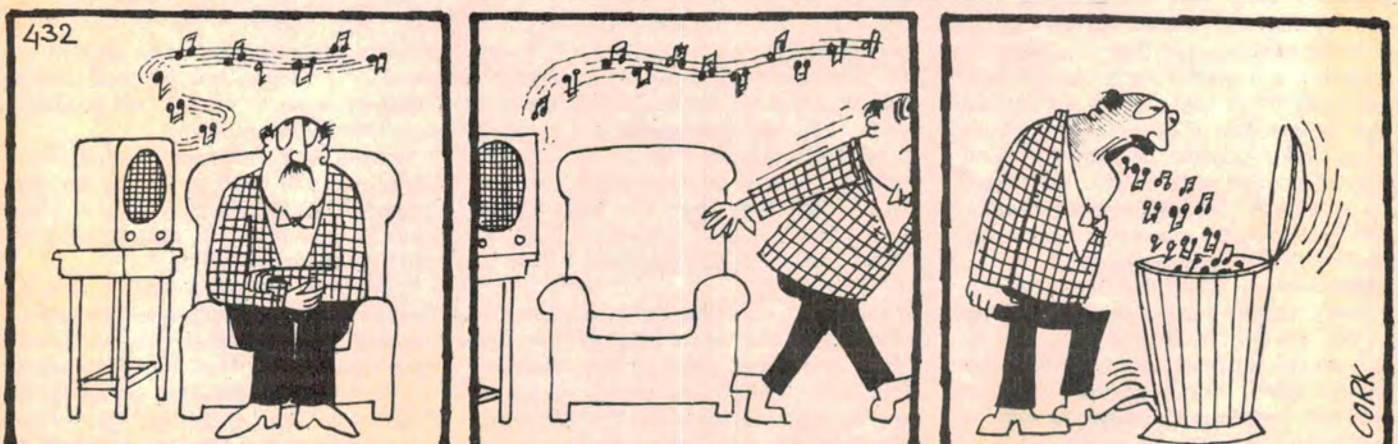


Fig. 17



IC5. The latter now reacts to the available HHHH input by sending pin 15 low, this pulse becoming the down beat for the IC5 cycle of line pulses.

(Note that there is an active output pulse on pin 15 of IC4 at this moment, as this pin returns to high. This pulse is so short, however, that the energy content is insufficient to activate the instrument simulator.)

This low pulse on pin 15, IC5, also appears, inverted by IC7c, at the input to IC6a (pin 3), but does not have any additional effect because it appears at this point simply as a continuation of the pulse which began on pin 15 of IC4. After IC5 has been through a full cycle of 16 beats and pin 15 goes low again, IC6 switches again and enables IC4.

The "start-stop" and "3/4" actions are the same as described before but require more connections to control more ICs. The circuit also shows the options suggested for the previous circuit, the down beat indicator and the start-stop touch switch.

The diode matrix represents the "recipe book" of the rhythm unit. It is not the intention of this article to provide the reader with a complete list of recipes. The reason for experimenting with rhythm units is to develop new and interesting patterns. The way to do this is to listen to a record, analyse the rhythm and to try to simulate it in the diode matrix. This takes time and patience but is great fun, especially when one plays an instrument like a piano or an organ.

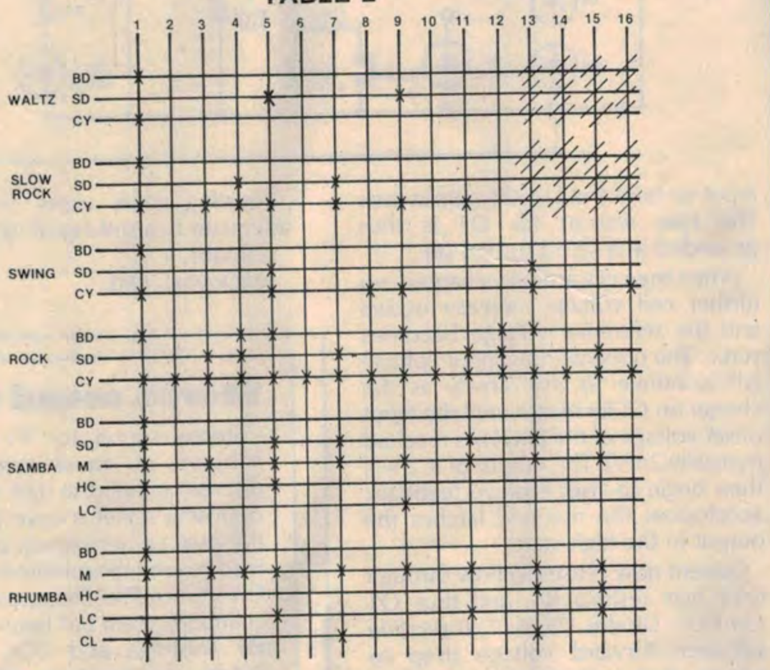
In order to set up the rhythm unit with some test patterns, Table 2 gives some recipes for a 16-beat counter. Fig. 17 shows the actual diode matrix for the "rhumba" pattern in Table 2.

Rhythm selector switches are required to connect certain diode matrices to instrument simulators. For each rhythm, one multi-pole switch is needed. As explained in Part 1, such switches are difficult to buy, and expensive. For that reason it was decided to use single pole switches which control type 4016 quadruple bilateral switches. These ICs are readily available and as the four CMOS gates in each IC are individually accessible, total freedom is obtained in determination of the number of poles per selector switch.

Fig. 18 shows the "waltz" selector switch. The number of switch elements is equal to the number of instrument simulators to be activated (plus one acting as the 3/4 switch in this case). With the selector switch in the "off" position, the control inputs of the CMOS switches are kept low, causing them to be turned off. In the "on" position, the control inputs go high and each CMOS switch element now connects the appropriate diode pattern to the instrument simulators.

Continued next month

TABLE 2



BD : BASS DRUM SD : SNARE DRUM CY : CYMBALS M : MARACAS HC : HIGH CONGA LC : LOW CONGA CL : CLAVES

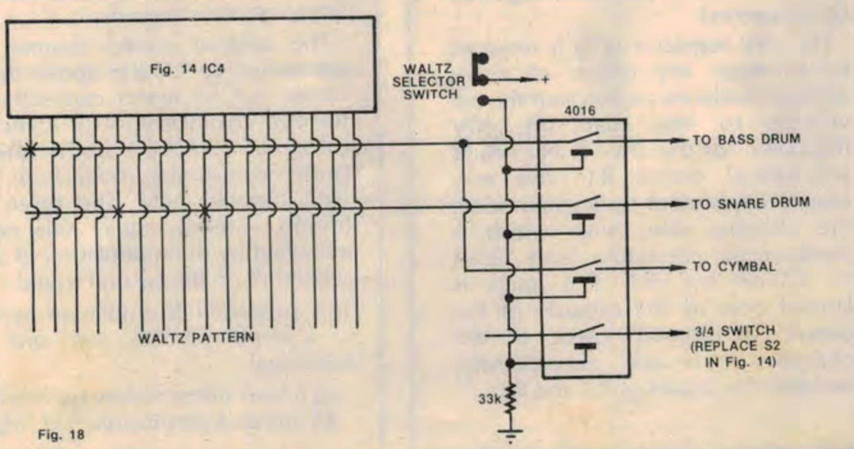


Fig. 18