Introduction to Direct Digital Synthesis.

Yves Marcoux, 1984-02-26.

# INTRODUCTION



Ever since musicians have known about computers, the many possibilities of using them in the musical domain have been explored. Already in 1959, Hiller and Isaacson (1) have experimented in the field of composition with the help of computers.

In the domain of sound generation, computers can be used for controlling electronic music equipment. Groups of adjustments of the equipment can be programed by the musician before the performance, and then executed rapidly by the computer during the performance, thus allowing a more elaborate use of the equipment. Computers can also, for example, remember sequences of diverse adjustments (like, for instance, notes), and then execute (play) them repeatedly, with or without programmed modifications, while a live performer can simultaneously vary other adjustments. We will call that type of tasks performed by the computer "macroscopic-time-level-control" of the generated sound.

Now, of course, those tasks can be programmed up to any degree of complexity (as long as it does not exceed the capacity of the computer used), and can include, among other things, automatic composition.

Another way of using the computer in sound generation has been developed around 1963 by Mathews (2). The technique is called Direct Digital Synthesis (DDS), and allows the computer to realize the sound synthesis itself, i.e. totally control the shape of the sound wave in its tiniest details, and that, for each and every one of its cycles. Theoretically, any audible sound wave can be synthesized by DDS. By analogy, we could say it gives the musician the same control over the sound wave as if he would hand-engrave a gramophone record.

Obviously, however, the musician cannot explicitly and exactly specify the wave form for the whole of a work of normal duration. This would be like hand-engraving the whole work on a record. So, in practice, the computer must be programed to generate the sound wave "on its own" at the microscopic-time-level. However, there are in theory infinitely many ways in which a computer can be programmed to generate a sound wave on its own. Moreover, any desired change in the computer programmation can be accomplished by "software" modifications, as opposed to "hardware" ones like electronic circuit modification, recabling, etc. So DDS remains one of the most, probably the most flexible sound synthesis method known.

The purpose of this article is to explain how DDS can be achieved, to present some variations of the basic technique that can be used, and to give a few examples of how a computer can be programed for DDS and of how macroscopic-time-level-control of the synthesized sound can be done in combination with DDS.

The discussion is held at a general level and many theoretical points are left undiscussed, and many examples are simplified.

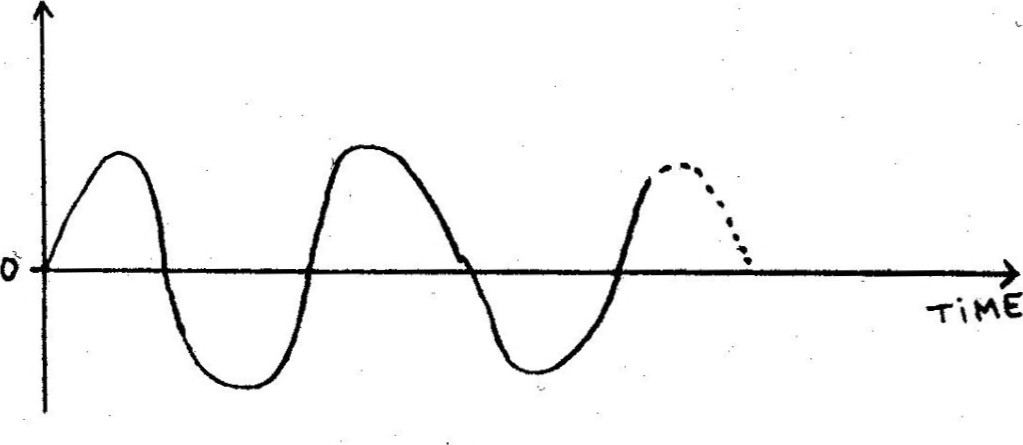
The reader is assumed to have good knowledge of acoustics and analog synthesis, but little of computers.

The abbreviation MTLC is used in the text and means Macroscopic Time Level Control, and refers to a group of techniques (that can be programmed on a computer) for controlling the generated sound that do not involve the sound synthesis itself, as presented above.

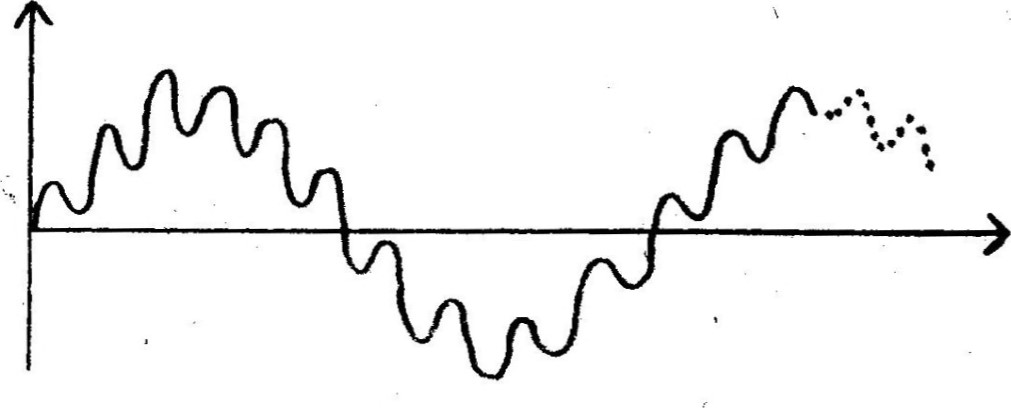
# The theory

## Sound and its analog treatment

Sound is a variation of air pressure in time at a given point of space. It is customary to represent sound graphically as the intensity of the air pressure plotted against time. For example, a sinusoidal sound wave is represented by the following:

Pressure

Since the air pressure at any given point of space can have only one value at any time, any combination of sounds reaching that point, regardless of the number or position of the sources, can be entirely specified by giving the description of only one sound wave. For example, the following wave represents a combination of two sinusoidal waves with different frequencies "played" simultaneously:



When presented such a wave, the human ear will "split" the two components and we will actually (in most cases) hear two different notes. However, it is a one and only resulting wave that has hit the eardrum.

With electrical analog sound treatment (like radio or analog recording), the sound wave is mechanically transformed into an electrical wave of the same shape (by a microphone) and this one is directly used to control other electrical elements (for instance, an amplifier). At the end of the process, the resulting electrical wave is converted to sound by a loudspeaker. (Fig. 1.)

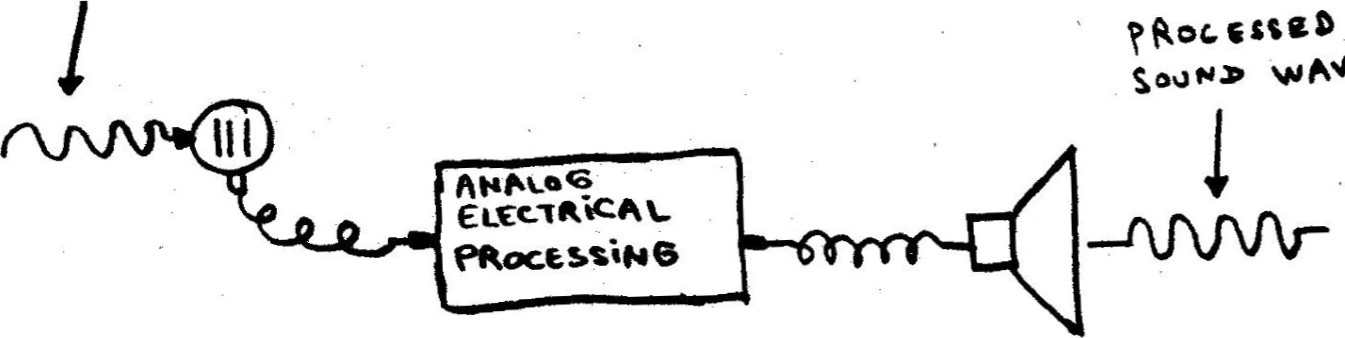
1.2- Digital treatment of sound

Now, there exists another method of treating sound electrically. It is the "digital" method. Although it can seem unnecessarily complicated at first glance, it can have advantages in some applications. We will explain it by using the example of a digital recording system.

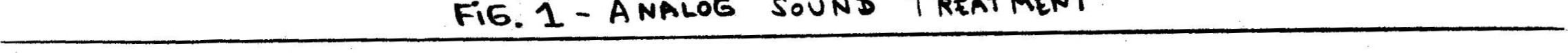
As in the analog method, the sound wave is first converted to an electrical wave (of the same shape) by a microphone. However, instead of being then used directly, this wave is converted into a series of numbers by a process called "digitalization" or, more simply, "sampling".

# .2.1- The sampling process

At predetermined and equally spaced points of time, the wave is inspected (measured), and its "height" is converted into a number (or "sample") according to a predetermined scale like, for example, -1 for the lowest possible height and +1 for the highest. The operation of converting the wave height into a number is done by a fairly common electronic component called an Analog-to-Digital Converter (ADC).



wAVE



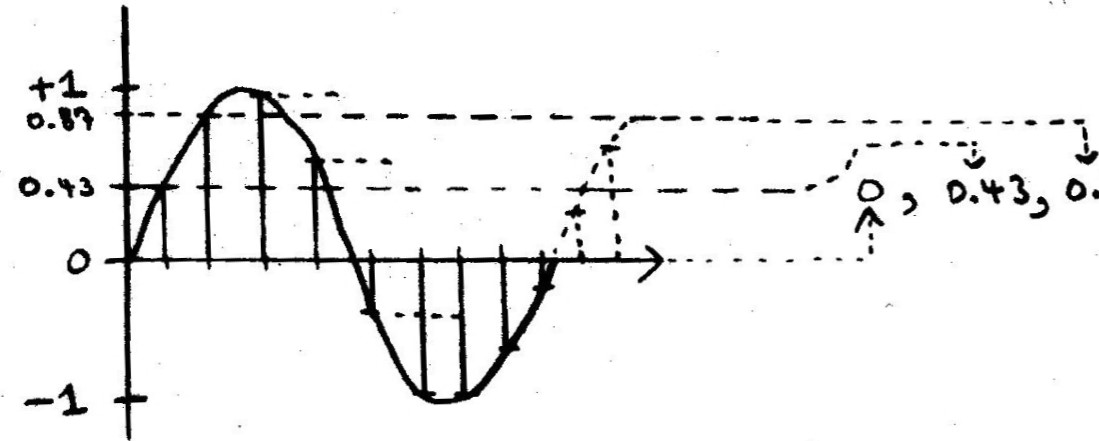
ΜξΝΤ

Α

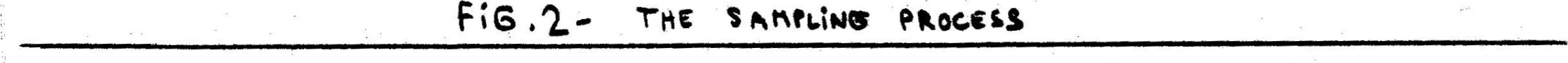
ΝΑΙ-06

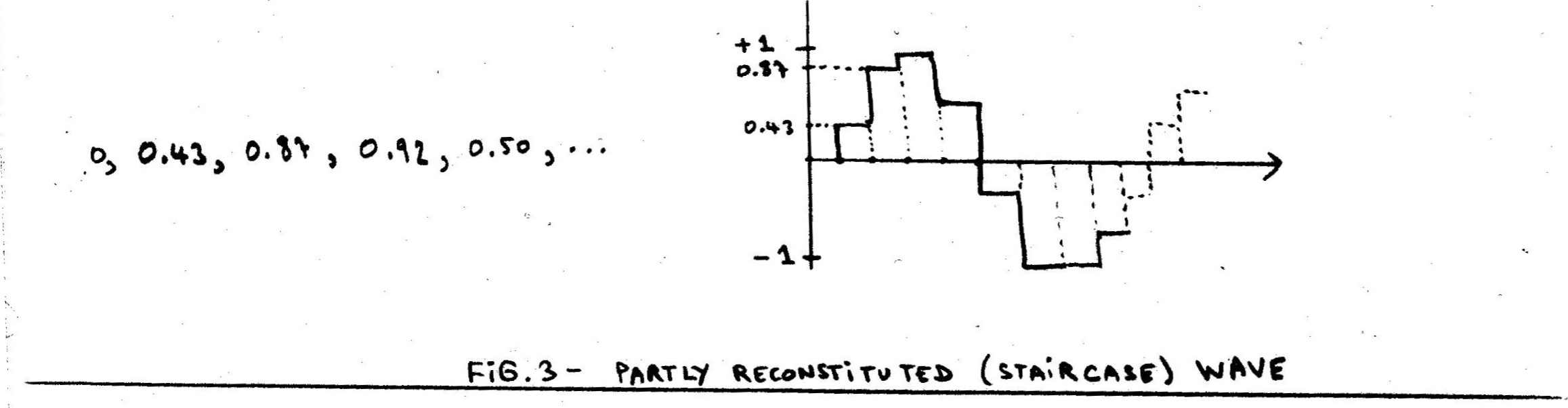
SoON>

-Τ¯ΑεΑτ

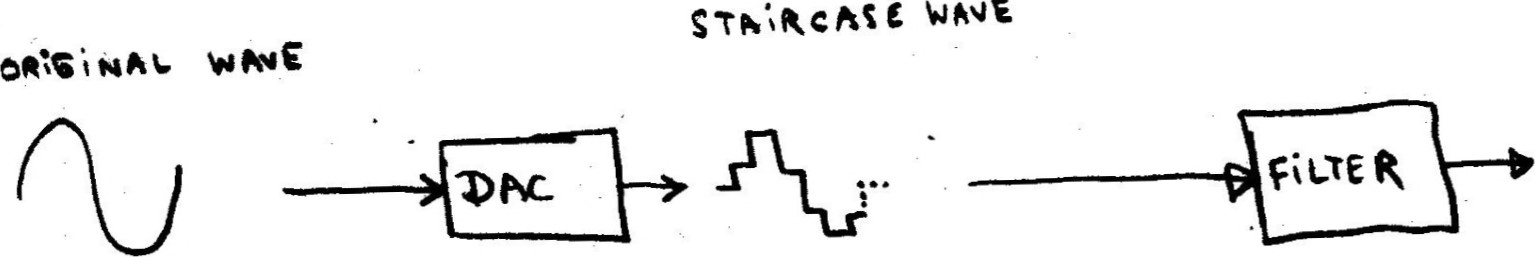
ο.ΙΆ, 6.92 -0.31

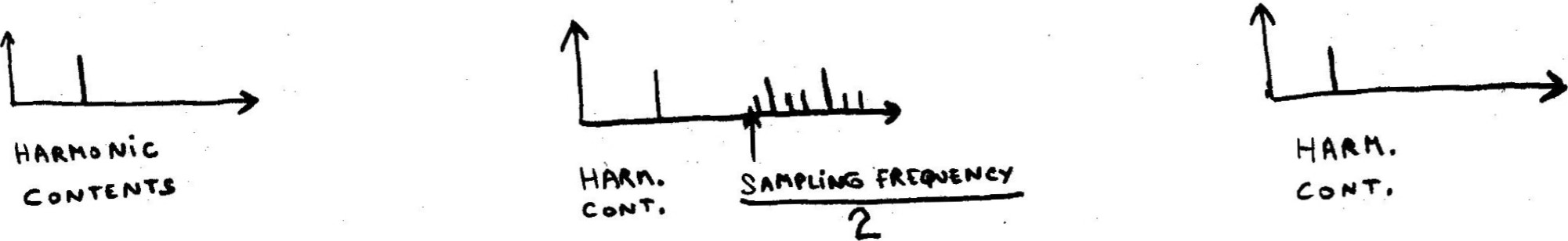


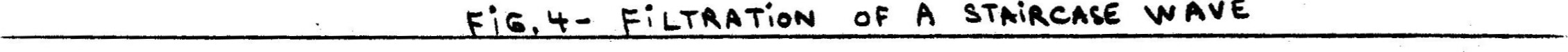


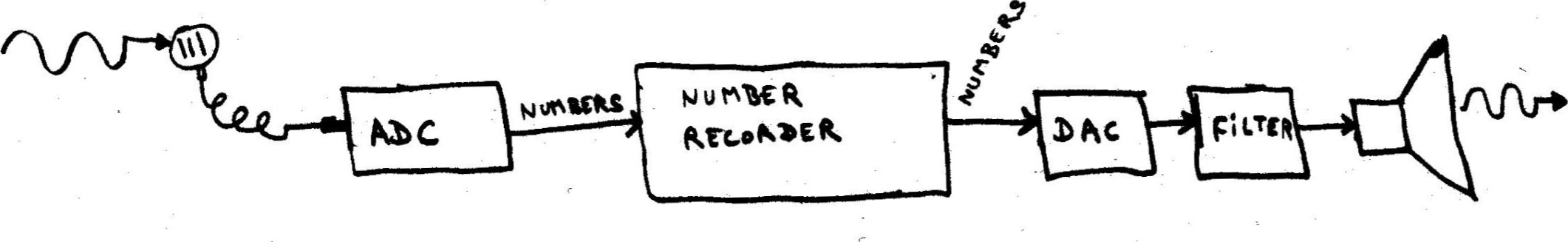


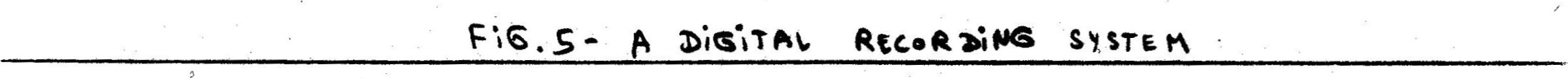
w AVK Γ-ΐΙ-Τξ











At each time the wave is inspected, a new number is generated. Between the times the wave is inspected, no number is generated; the wave is in fact ignored. Fig. 2 illustrates the sampling process.

Then, nothing more is done with the wave. All the further processing to be done by the system is performed on the numbers produced during the sampling process.

Now, in the case of our example, digital recording, the processing to be done is to store those numbers on an adequate medium (for example, magnetic tape) so that they can be retrieved later. We will not describe how the storing and retrieving of the numbers can be done. We will just assume-. that playback time has come, and that the samples are successfully retrieved from the medium at the same rate at which they were taken during the sampling process. Now, we don't have the original electrical wave anymore. So, we are faced with the task of reconstructing it from the samples we have.

Here, we have a problem.

1.2.2- The reconstitution process

Indeed, we know the height of the wave at precise points of time, but we have no idea of what it looked like between those points of time.

Well, there is at least one thing we can do with our samples, and it is to convert them into electrical "heights". In effect, there exist electronic components as common as ADC's, and which perform exactly the inverse function. They are called DAC, for Digital -to-Analog Converter. The DAC can be set so that it will use the same conversion scale as the ADC.

So, each time we receive a sample from the medium, we can at least send it to the DAC. This one will immediately set its electrical output at the corresponding height, and hold it there until we send it a new number. Now because we don't know what the height of the original wave was until the next sample, we can do no better than wait for it and send it to the DAC as soon as we receive it.

If we look graphically at the output of the DAC through time, we see that it forms a "staircase"-like wave that "imitates" more or less accurately the original one. But it has not exactly the same shape. It really seems that we cannot reproduce exactly the original wave with only the samples as a starting point. (FRY. 3)

Fortunately, it has been proved mathematically in the 1920's that such a staircase-wave reconstructed from samples has exactly the same harmonic contents as the original wave, up to half the sampling frequency, i .e. the rate at which the samples are taken, expressed in samples per second (sps). For example, if we sample at 10000 sps, the staircase-wave constructed from the samples will have exactly the same harmonics as the original wave, up to 5000 Hz. From 5000 Hz upwards, the harmonic contents will generally differ. This mathematical fact is known as the "sampling theorem".

Now, assume the original wave has no harmonics above half the sampling frequency. If we remove from the staircase-wave, by filtration, the harmonics above that frequency, we are left with exactly the harmonic contents of the original wave, hence, the same wave form. (A filter is an electronic component that removes certain harmonics from an electrical wave that goes through it.)

So, in our example, if we take the output of the DAC and filter it properly, we are able to produce an electrical wave that has exactly the same shape as the original one, provided that the latter has no harmonics above half the sampling frequency. If we then take that wave, and send it to a loudspeaker, we can say that our digital recording system is able to perfectly

reproduce any sound wave that has no harmonics above half the sampling frequency. ( . S)

The process of sending samples to a DAC at the same frequency at which they were taken, and of removing from the output of the DAC the harmonics located above half that frequency is known as the "reconstitution process".

1. .2.3- Making the system "perfect"

Now, if we choose the sampling frequency to be 40000 sps, all the hamonics contained in the original sound wave up to 20000 Hz will be intactly reproduced by our system. But, it happens that 20000 Hz is .just about the highest frequency that can be perceived by humans. So, we can in fact say that our system would perfectly reproduce any audible sound wave. In other words, if a sound wave contains harmonics that could not be reproduced by our system, then those harmonics could not be heard by humans anyway!

In the context of sound processing in general, and not just of digital recording, the sampling theorem implies that any audible sound wave can be entirely and exactly described by samples (numbers) produced at a rate of 40000 sps, and then actually made to sound by the reconstitution process.

It must now be said that the numbers used in a digital sound system are themselves coded into electrical signals, but those signals have so little in common with the corresponding electrical wave, that the sampling and reconstitution processes must indeed be regarded as conversion processes.

1.3- Using a computer to generate digitally coded sound

The processing of digitally coded sound involves, as we can imagine, a lot of number manipulations. Computers, because of their speed and ability to deal with numbers, are prime candidates for use inside a digital sound system. In fact, any fast enough computer programed appropriately, and connected to the necessary equipment can be turned into a digital sound recorder.

In this case, the computer would be programed to store the samples received from the ADC and then, when desired, send them to the DAC for reconstitution.

But could not we also program the computer so that it would send to the DAC samples not coming from the ADC, but other samples, describing a sound wave that has never been sound before? A sound wave that would be generated by the computer?

Well, computers are extremely powerful and flexible devices, and surely they can be programed to produce whatever sequence of numbers we can possibly think of. Now, we have said earlier that any audible sound wave can be entirely and exactly described by samples produced at-the rate of 40000 sps, and then actually made to sound by the reconstitution process. So it follows that if we have a fast enough computer, we can, through proper programming and with reconstitution equipment, make it produce any conceivable audible sound wave. That technique is called Direct Digital Synthesis (DDS).

1.4- The programming problem

Let us consider the problem of programing a computer for DDS from a general point of view.

In analog synthesis techniques, naturally oscillating elements (oscillators) produce waves that are combined and processed to yield an actual sound.

Those t'eo operations are performed by electronic components, which of course are limited in number in any given synthesizer. The freedom of the musician in determining the actual sound is thus limited to controlling those electronic components by, for example, setting knobs, depressing keys, connecting cables.

In DDS, however, the actual sound is not determined by such controls. It is determined by each and every sample that is sent to the DAC. So in fact, in order for the musician to totally specify the actual sound, he would have to decide the value of each and every one of the samples to be used. Now, we don't think anybody would be interested in entering many thousands of numbers in a computer just to produce one second of sound, so we can in practice reject that possibility.

SG S t s AN'

The only acceptable solution is to program the computer so that it will be able, at the "microscopic-time-level” involved in the individual samples generation, to "go on its own" and generate the samples without being given them. We must give the computer a "scheme" according to which it can generate the samples. The scheme must decide enough about the generated wave fom so that the amount of human actions to specify the actual sound falls into the reasonable. We will call such a scheme a Sample Generation Scheme (SGS). When refering to an actual cmputer program real izing an SGS, we will talk about Programed SGS (PSGS).

Like any instructions to be given to a computer, PSGS t s must be programed in a form understandable by this one, i.e. a "programing language", like BASIC or FORTRAN. Now, it is not likely that musicians will directly program SGS's in ordinary data processing programing languages. But, as we will see later, through the use of specialized high-level programing languages, like MUSIC V, they can be given the possibility to indirectly program interesting and quite varied SGS’s. Sec (3)

There are in theory infinitely many SGS's that can be programed on a computer. One of the simplest ones is for example to generate the samples at random. This would produce white noise. We will give later more examples of SGS’s.

1.4.2- PSGS-musician communication

Now, regardless of what SGS is used, it need not decide everything about the sound Wave. Just like it is convenient that an oscillator does not produce always a wave of the same frequency, but that this one can be specified not while building the oscillator, but at "performance time", it can be convenient that some parameters of the wave generated by our SGS be decided not while programming it, but at "performance time".

So, there must be communication between the musician and the PSGS at performance time.

For the example of frequency specification at performance time, the communication can be realized by connecting a piano-type electrical keyboard to the computer in such a way that the PSGS will be able to know at all times which note(s) is (are) depressed. (That information is (once more!) coded into numbers.) The musician can then play on the keyboard as on an ordinary one, and the PSGS (if appropriately programmed, of course) will change the frequency of the sound wave it generates according to what the musician plays.

There are many other types of PSGS-musician communication that can be used. For example, potentiometers, joysticks, etc. can be connected to the computer. In all cases, the information of the position of the device is sent to the

PSGS numerically coded.

2- The practice

# 2.1- Practical considerations

TWO FAZTS

We must immediately mention that have been withheld so far for simplicity's sake, and which place slight restrictions on the sound waves that can be generated by DDS.

The first one is the following: when a computer wants to send a number to a DAC, it must do so with a limited precision. By that, we mean that all numbers must be truncated to a limited number of "digits after the decimal point", to make it simple. For example, the value 1/3 (which is in fact 0.333. . . with an infinite number of digits "3" after the decimal point) would be truncated to, for example, 0.33333. So this in fact limits the precision with which the computer can specify the value of a sample, thus the height of the wave to be generated. However, this limitation has in practice little significance, and will be ignored in the rest of this article.

The second one is that in practice, perfect filtration is impossible. This is however generally not regarded as very significant, and will also be ignored in the rest of this article. 

2.1 .1- Lower sampling frequency

DDS is possible even if the computer used is not fast enough to produce

40000 samples per second. First of all, we can accept to limit the harmonic contents of our sound to less than 20000 Hz, let's say 15000, and we suddenly need to produce only 30000 samples per second. That's a 25% reduction

on the required computer speed. Of course, if we change our sample generation frequency, we must adjust the filter in the reconstitution process so that it still cuts harmonics located above half the new frequency. A sampling frequency of 30000 has apparently been used satisfactorily by

Mathews (3).

2.1 .2- Deferred sound performance

Second, we can make the computer generate the samples at its own (maximum) speed, and record them (like in digital recording). When all the samples have been generated and recorded, we send them at the desired speed to a DAC for reconstitution. It is then only that the sound is performed. With this technique, the performance of the sound is not "live". But it has the advantage that it can be used with as slow a computer as "desired". And, of course, slower computers are cheaper.

Note that deferred sound performance does not necessarily prohibit the musician from playing at a natural tempo. In effect, through MTLC techniques, the "melody " (succession of notes played) can be recorded, and then sent to the computer (PSGS) at the required (lower) speed.

2.2- Some types of SGS’s

Let us consider some types of SGS’s.

2.2.1- Physical system simulation

One of the first SGS’s that come naturally to the mind is that of simulating physical systems. With such a scheme, the computer would be given the

parameters of a physical system, and then simulate its reaction to outside stimuli. Then the resulting vibration induced at a particular point of the system would be measured at regular time interval and used to generate the samples. For example, the motion of a pinched string could be simulated, and its height at a precise point used to generate samples. One of the performance time parameters in this SGS could be the tension in the string. It could be, for example, specified by the musician by playing notes on a keyboard connected to the computer.

There are many physical systems that can be simulated by computers, and not only mechanical ones, so the possibilities in this area seem numerous. However, computer simulation involves an enormous amount of computations, even for simple systems and even with the fastest computers available today, it can hardly be done for live sound performance. It can however be done in deferred sound performance.

2.2.2- Processing of pre-recorded sound

Yet another SGS that can be used is the processing of already recorded sound. For example, one note of an instrument can be (digitally) recorded by the computer, and then some parameters of that note (pitch, duration, envelope, etc.) varied to produce new sounds. This method is not as simple as it may sound, because the different parameters of the note have to be “recognized” by the computer in order to be modified. As far as we know, it is already today used in live sound performance.

2.2.3- Algorithmic SGS t s

Another type of SGS's can be achieved by generating the samples with mathematical formulas and algorithms. (An algorithm is a "recipe" to be followed and that gives the order in which the mathematical formulas must be applied.) Because computers are so good at mathematical formulas and "recipes" algorithmic SGS’s can be programmed easily and efficiently. Some formulas can, without simulating them, "emulate" complex physical systems (1ike electronic oscillators), and thus generate complex sounds without too many computations.

Mathematical formulas and algorithms are in essence what makes up the "native language" of computers, so one could argue that any SGS, when programmed into a computer, is expressed as an algorithmic SGS. Still, we believe that a less strict classification of the SGS’s, like the one suggested above, is useful.

Appendix I gives an example of an algorithmic SGS from which a programing language similar to MUSIC V can be derived.

3- The totally computerized music machine

There is nothing that prohibits more than one PSGS to be "run" simultaneously on the same computer. Very common techniques in computer science permit that. In that case, the control of the PSGS’s can be, of course, "programmed" (on the same computer) by MTLC techniques (sequencing, automatic composition, etc.), just as if the PSGS’s were external to the computer controlling them.

However, the enormous advantage of having the whole thing run on the same computer is that the connections between controlling and controlled elements do not have to be physical (e.g. cables), but are simply programmed. The same flexibility advantage that DDS has over analog synthesis is then obtained at the MTLC level.

The power of a totally computerized sound generation system has long fascinated the researchers in the domain. We believe that Max Mathews, in 1970, has quite justly called those systems not merely synthesizers, but "music machines" (3) .

Conclusion

In this article, we have explained how DDS is realizable, showed that SGS I s are necessary to achieve it and given examples of those.

We have also shown that PSGS's must generally communicate with the musician at performance time, and given examples Of how that communication can take place.

We have also tried, to the best of our knowledge, to relate the presented notions to actual realizations in the domain.

We hope to have given the reader an idea of the possibilities of DDS.

