Fast Fourier Transforms

You may have heard of Fourier Analysis, but more often than not explanations of what this is all about are drowned in mathematics. E.J. Hughson describes how it's done electronically.

MUCH OF ELECTRONICS is concerned with the processing of signals of some sort or another. It is only natural then, that a lot of effort has gone into analysing these signals. On one hand one must know certain basic things about the signals in order to be able to build useful circuits. On the other hand, investigating signal properties with no particular applications in mind, has led to various useful results that later helped to simplify, improve or introduce new circuit designs. The field of study concerning signals is known, naturally enough, as "Signal Analysis".

In order to go deeply into some of the theory in this field, some pretty heavy math must be employed. However, it is quite easy to understand the majority of the material intuitively, and besides, that's a much more entertaining approach.

Think of a Signal . . .

How do most of us think of signals? Probably "signals" conjures up images of a scope with a waveform on it. Let's use this waveform as an example — suppose it's a 1kHz triange wave. What characteristics does this waveform have? It is a voltage (say) varying up and down periodically, thus it has an "instantaneous amplitude" at any instant in time. This is what we see on the scope, an amplitude versus time graph. We can also say that the waveform has a characteristic we call frequency. Most of us use the term frequency to mean the basic frequency of repetition of the entire waveform. Why this distinction? Here's where a theoretical concept must be just accepted if we're not to get submerged in abstraction.

Fourier Analysis

It is convenient to think of a *sine* wave as the "purest" waveform, and use this kind of wave as a basis for study of other waveforms. It has been found possible to make any other kind of waveform from a combination (sum) of sinusoidal waves of various frequencies and amplitudes. This is analogous to being able to combine the three basic colours of light, green, red, blue to form other colours.

In fact we have cheated a little bit; we should correct the above to say that any kind of waveform can be made from combinations of sine *and cosine* waves of various frequencies and amplitudes, a cosine wave being simply a sine wave but one quarter wave ahead.

Ok, so what? The next step is to introduce a graph of amplitude versus frequency. Figure 2 is an example in which we plot the "frequency content" of a sine wave of amplitude 1 and frequency 1kHz. There is only one point on the graph, because as we said before, a sine wave is considered to be "pure" or only one frequency.

So how about our triangle wave? What does its frequency content look like? Figure 3 shows that the frequency content is quite complex.

The graph shows that htere is a large content of the fundamental frequency, with decreasing content of odd order harmonics.

The process of converting the "time" waveform to the "frequency" graph is called the Fourier Transform. The reverse process is called the Inverse Fourier Transform.

In the case of a repetitive waveform (such as the triangle wave) the Fourier Transform yields a frequency content graph which has non-zero points only at multiples of the fundamental frequency. Thus, a series of numbers may be used rather than a graph to represent this information. For the triangle wave, the series is:

.81 x (1kHz sine — .09 x 3kHz sine) + .032 x (5kHz sine) — .017 (7kHz sine) + . . .

For a 1kHz, ±1V square wave the series is:

1.27 x (1kHz sine) + 424 x (3kHz sine) + .255 x (5 kHz sine + . . .

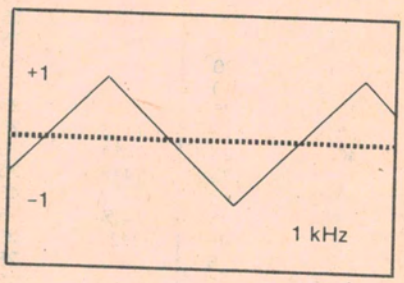


Fig. 1. Scope trace showing triangle wave

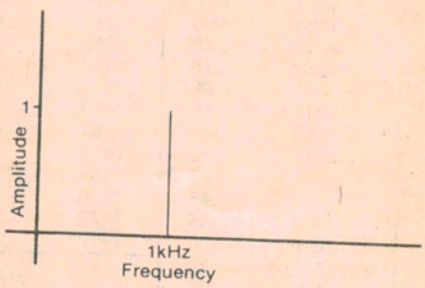


Fig. 2. Amplitude vs. frequency plot of 1KHz sine wave

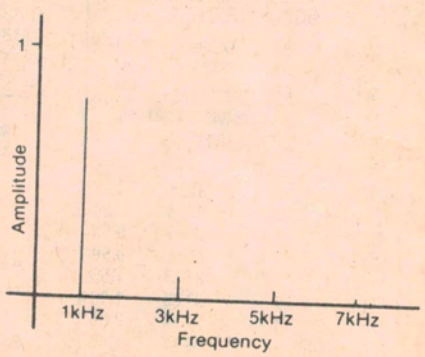


Fig. 3. Amplitude vs. frequency plot for triangle wave.

On the other hand, you are no doubt already familiar with frequency plots of noise, and particularly audio equipment response curves, which are nothing more than the frequency content graphs of the output with "all frequencies" fed in. (Fig. 4.) Note that the frequency plots in this example are continuous rather than just the odd point here and there.

Doing It

A picture of actually doing the transform is shown in Fig. 5. The "transformer" could be a person with a piece of paper working out the graph or more usefully a machine doing the work. Suppose a computer was used to do the task on an input waveform, how would it do it?

Automatic Transform

If an analog waveform is sampled at regular intervals, we get what is called a discrete time series — discrete because it is a series of separate points and time series because we have something changing with time. Figure 6 shows a sampled sine wave displayed on an oscilloscope.

If we were to measure the level of each of the points we would get a series of numbers. If we do this electronically using analog-to-digital conversion we get a series of digital numbers representing the discrete time series. OK so far? It is this set of digital samples which a Fourier Transform (or "Discrete" FT in this case) takes and turns into information directly showing the frequency or harmonic content of all signals which make up the original time series. The technique shows any components from DC to half the sampling frequency. (It is not possible to obtain any frequencies higher than this since it would contradict a fundamental rule concerning sampled waveforms, established by Nyquist.)

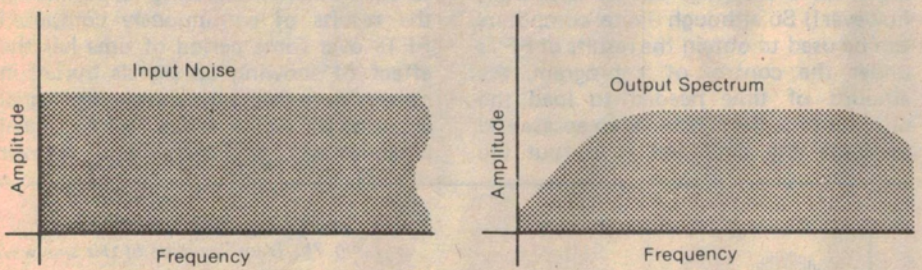


Fig. 4. Noisy amplitude vs. frequency spectra

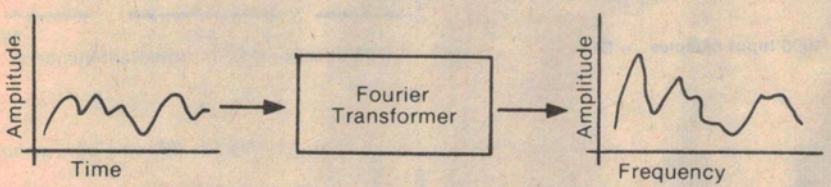


Fig. 5. The transform process

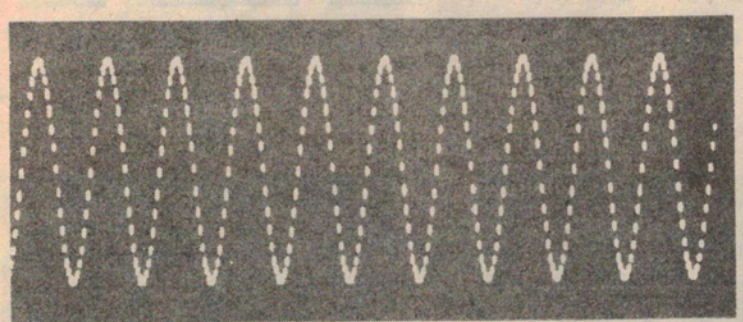


Fig. 6. Analog sine wave sampled at intervals

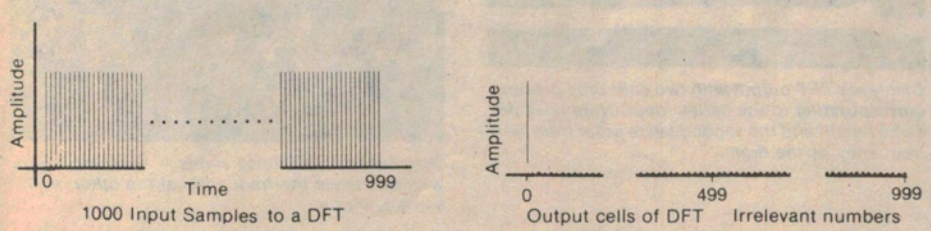


Fig. 7a. Transforming a DC voltage

Adding New Frequencies And Filtering

If the output numbers undergo an inverse DFT we get a series of numbers outputted which represent the original waveform.

By taking a waveform and analysing it using a DFT, then performing an inverse DFT on the output we can arrive back at the original waveform. A filter can be made by performing an inverse DFT only on those numbers representing the frequencies which are required.

Similarly, by adding numbers to represent new frequencies before performing an inverse transform extra frequencies will be present in the output time series (and after digital-analog conversion, in the output waveform).

The DFT does not work on analog or continuous information: only on a set of numbers representing the instantaneous values of a portion of a waveform. The result is a set of numbers corresponding to the frequency content of the waveform. Not only does DFT give us each frequency present in the original waveform, it also gives the relative phase and amplitude of each frequency component.

By performing a power calculation on the output frequencies a power spectrum may be obtained. Of course the more numbers or samples which are input to the DFT, the more information is available at the output. However, for a fixed set of numbers is outputted.

Interpreting The Numbers on the Output

To illustrate how outputted numbers are interpreted consider a DFT performed on a portion of a time series containing 100 samples. The 1000 numbers inputted will have various arithmetic operations performed on them and 1000 numbers (known as frequency cells) will be outputted. Of these 1000 cells only the first 500, representing the frequency range, will have any real meaning.

For example, suppose all input numbers representing samples from an anolog-to-digital converter are the same. (This would mean that a constant DC voltage would have had to have been applied to the converter.) Of the 1000 numbers obtained by the forward transform, only the first would have a value other than zero, since this first number is reserved for the DC content of the input series and all the energy of the input is in the form of DC. (See Fig 7a).

Suppose now the output of an analog-to-digital converter is being sampled at 1000 samples/sec, also suppose a sine wave of 1Hz is applied to the input of the converter. One thousand numerical samples or one second's worth of data is collected. If these 1000 numbers are used as the input of a Fourier Transform, then of the 500 numbers output, the first will have zero value (DC) but the second, reserved for frequency of 1Hz, will have maximum value (Fig 7b). All others will have zero value also.

If the frequency of the sine wave inputted to the converter is now increased to 2Hz and the 1000 samples at 1000/sec are collected, the Fourier Transform processor output will consist of zeroes in all 500 numbers except the third corresponding to 2Hz (Fig 7c). The output numbers are the cells, cell 0 to cell 499 in this case being reserved for frequencies of 0 (DC) to 499Hz. Figure 7 gives a graphic representation of these inputs and outputs. (NB, since the output cells are numbered starting from zero so also are the input samples, for clarity.)

Cell Number And Frequency

The example given above assumes a sampling frequency of one thousand per second so that with a 1000 point transformer the cell numbers automatically correspond to the frequencies they represent in the input time series. It is of course, not always practical to have the sampling frequency tied to the number of samples in the DFT as rigorously as this. But it is a very simple matter to obtain the actual frequency to which a cell output corresponds. This is obtained by the following relationship:

$$\text{Frequency corresponding to Cell Number (1st cell = 0)} = \frac{\text{Cell No. x Sampling Frequency}}{\text{No. of points in FFT}}$$

The outputs depicted by Figure 7 are of course idealized. In practice slight errors will occur due to the finite number of bits used in the arithmetic of the calculation.

As so far discussed, the Discrete Fourier Transform both forward and reverse has been put in terms of numbers which are inputted, the calculation process and numbers outputted. The calculation process is very involved and tedious but could be carried out by a computer or even a hand calculator (if you had the time and patience). To perform a 1000 point transform, it would require over 2 million discrete calculations, tedious indeed!

Fast Transform

The Fash Fourier Transform technique is able to reduce the calculations of a similar size transform to about 22,000 which is a significant reduction in the number of calculations and hence the amount of computer time. (Still a little much for the average pocket calculator, however!) So although digital computers can be used to obtain the results of FFTs under the control of a program, the amount of time needed to load the samples into the machine, to access and compute the data and to output the results makes even a general purpose digital computer an impractical signal analyser.

For this reason, analysers using dedicated hardware capable of only performing FFTs are a far more practical proposition. Such analysers are capable of taking an analog wave form input performing analog-to-digital conversions on it, sampling accordingly, loading the desired number of samples into a dedicated FFT calculator and presenting the results to some display for presentation. See Fig. 8.

The instruments using FFT analysers usually come complete with accumulators and memories so that frequency spectra may be integrated and compared to each other. Integrating (or summing) the results of continuously computed FFTs over some period of time has the effect of showing up signals buried in noise. No matter how deeply the signal is buried the cell or cells which the signal occupies will eventually build up over

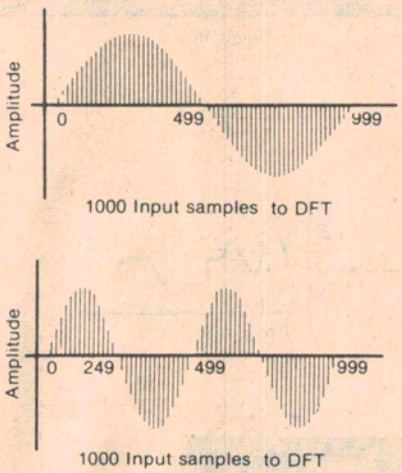


Fig. 7b. Transformation of 1Hz sine wave.

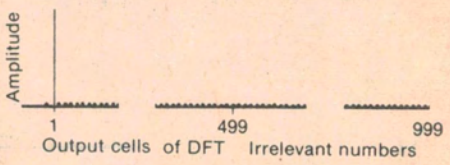
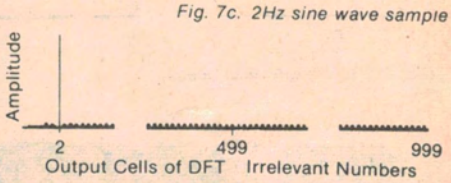
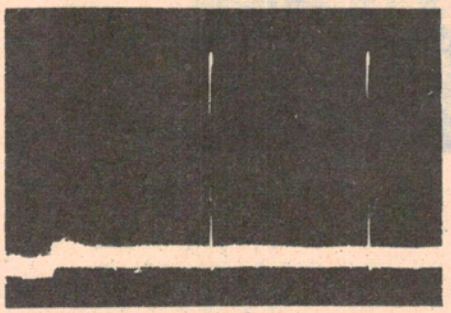
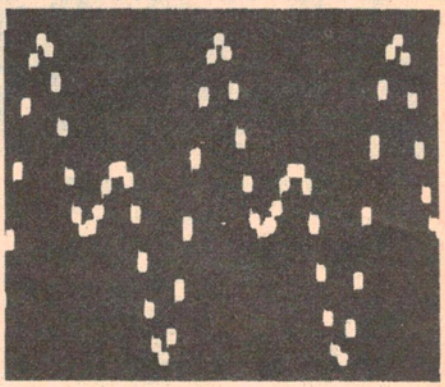


Fig. 7c. 2Hz sine wave sample





Displayed FFT output with two cells only present, corresponding to the series input. Note lines are same height and the second twice as far from zero frequency as the first.



Original sampled times series — two sine waves with one twice the frequency of the other, both same aplitude.

Fig. 9. Actual inputs and output of a FFT processor

all the other cells where the noise will be randomly distributed. This technique is now used, for example, in submarine detection where the noise from the vessel is discriminated over the sea noise by continuous integration of FFT results.

Figure 9 illustrates a sampled wave form consisting of 2 sine waves of equal amplitude and the display results of an FFT performed on a set of numerical samples taken from the time series. The display has its own amplitude graduations and the two lines represent the energy in the cells corresponding to the frequencies of the 2 sine waves. (Note no other lines appear as all other cell values are zero). The display is produced by continuously outputting the cell numbers from FFT result to a digital-to-analog converter and including the amplitude graduations.

The Mathematics

This article is not the place to consider the in depth mathematical theory necessary to fully understand the processes which form part of the Fourier transform. Numerous books and technical articles now exist on the subject. However the basic operation and its adaptability to digital hardware is quite easily understood.

One major constraint on an FFT processor is that the number of samples inputted to it cannot be varied completely. With most processors, the number of samples in fact have to be a power to 2, e.g., 32 or 54 or 256 or 1024. The more samples taken then the larger the range of frequencies which can be determined or alternatively the narrower the band width between cells. However, the calculation process takes longer. In practice sample blocks of 512, 1024, 2048 are amongst the most commonly chosen as these offer a compromise between frequency range and computation time.

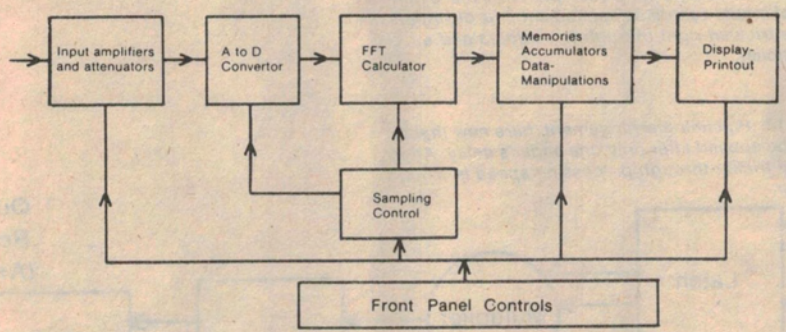


Fig. 8. Typical FFT Analyser basic functions

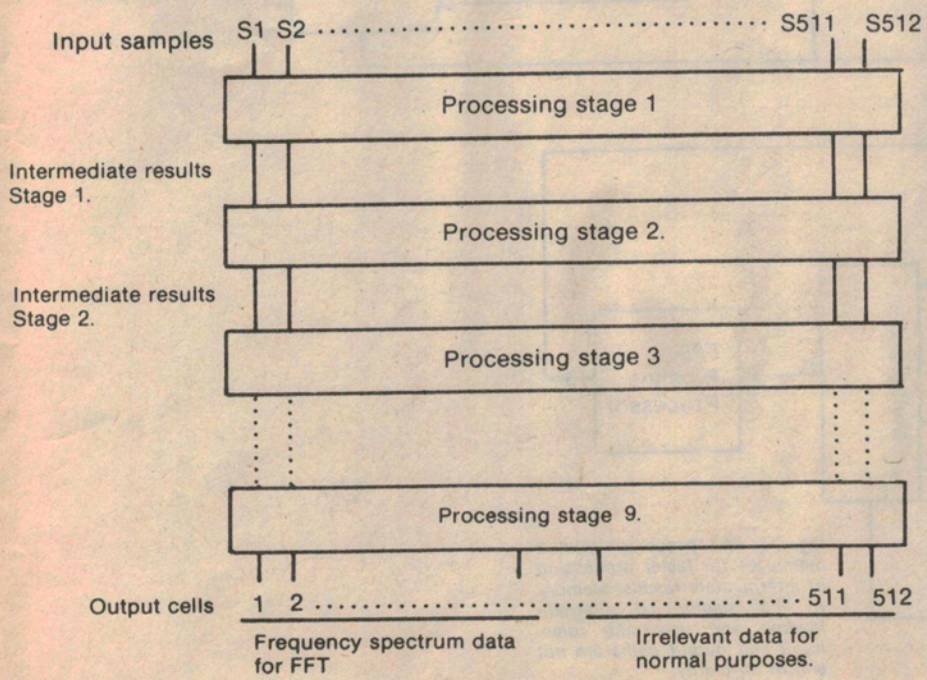


Fig. 10. FFT Processing stages for 512 point transform

Essentially in the case of a Forward Transform the samples from the time series are loaded into a buffer and combinations of samples are added and subtracted from each other, multiplied by trigonometrical values usually looked up from a Read Only Memory. This process is repeated using different combinations of samples and trigonometrical values. The number of processing stages is related to the number of samples, eg, if 2048 samples were inputted, then 11 processing stages are needed. (22^{11}=2048). If 512 samples were inputted, then 9 stages are required, etc. This is illustrated in outline by Fig. 10.

The advantage over the old 'conventional' method of computation is that with the conventional method the number of stages of calculation equals the number of samples. In a 512 point transform the process would be 512/9 or approximately fifty-five times shorter by using an FFT. In a continuous process where FFTs are being continuously computed, obviously a very real saving is made in terms of result presentation.

As mentioned earlier the FFT processing idea lends itself very easily to a dedicated machine and the idea of pipeline processing is used in most of these. Pipeline processing is used where a number of calculations in series are performed and where an unacceptably long delay result for the computing of an answer before the next inputs are applied. Figure 11 shows an arrangement where this is likely to happen.

In this example an adder precedes a multiplier followed by another adder. With no pipelining, no further inputs can be supplied until enough time is allowed for the results of previous input numbers to be stored away. However with a pipeline processor, (Fig. 12) latches are included in strategic places allowing sets of numbers to follow each other as though they were coming down a pipe. Thus after the first set of results have trickled through the latches, a much faster throughput of numbers will result.

This type of arrangement is very suitable in FFT processors since a large throughput of samples with much number crunching takes place.

Present & Future FFT Analysers.

Essentially an FFT processor (which is the heart of modern spectrum analysers, voice print identifiers, etc) usually consist of a memory which stores the samples and intermediate results and a processor which computes intermediate results. The total samples

are stored internally in the memory and when the FFT process begins, the samples are taken in pairs, arithmetically operated on to form intermediate results and stored ready for the next level of processing. Two memories are sometimes used where samples and results are alternately read from one memory through the pipeline processor into the the other memory, where they are ready for the next level of processing. This technique saves still more time in computing FFTs. See Figure 13.

This article has only touched onto the now very broad field of FFT processing technology. The approach lends itself easily to band shifting, frequency zoom effects and other features made relatively easy with a digital system.

Until the 1970s few people knew of Fourier Transforms. At best the term would evoke a feeling of something obscure, very mathematical and having few, if any practical applications. But in recent years a whole new world of applications has been unleashed.

FFT techniques are today used for a variety of applications including extraction of signals burried deep in noise, sonar processing, spectrum analysis of compex waveforms, voice print analysis and the digital synthesis of music. Research is finding still more uses, such as in the oral synthesizing interface of talking computers. Some day you may be able to phone a computer and hold an intelligent conversation with it, obtaining such things as account balances, travel reservations, etc, with tonal expressions no different from those of a helpful person!

With the advent of bubble memories and the ever decreasing size but increasing complexity of micro circuits, it appears that the FFT processing field will expand to a point where it will soon be a part of every day life, a truly big step forward from just 10 years ago when the technique was not even heard of.

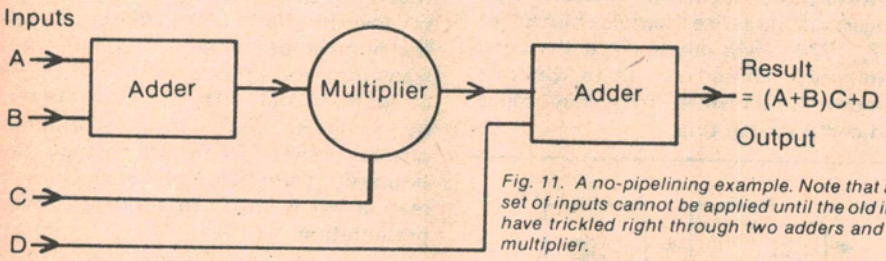
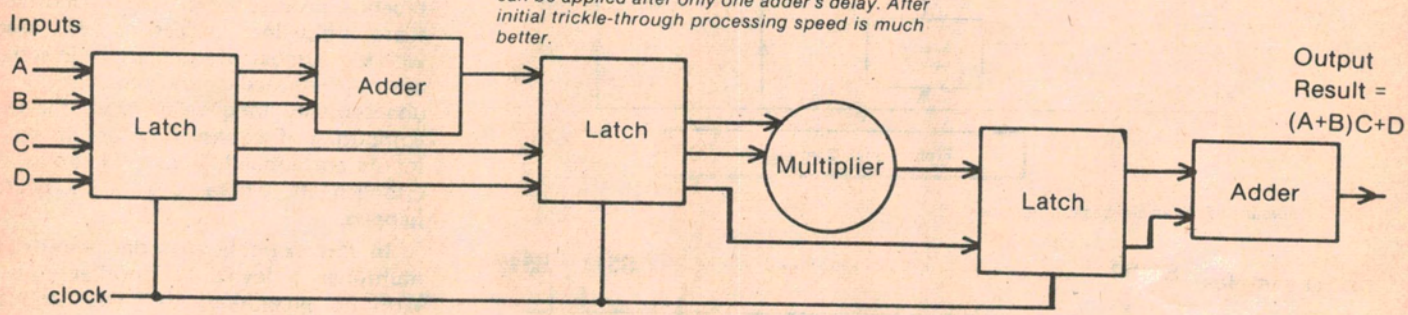


Fig. 11. A no-pipelining example. Note that a new set of inputs cannot be applied until the old inputs have trickled right through two adders and a multiplier.

Fig. 12. Pipelining arrangement, here new inputs can be applied after only one adder's delay. After initial trickle-through processing speed is much better.



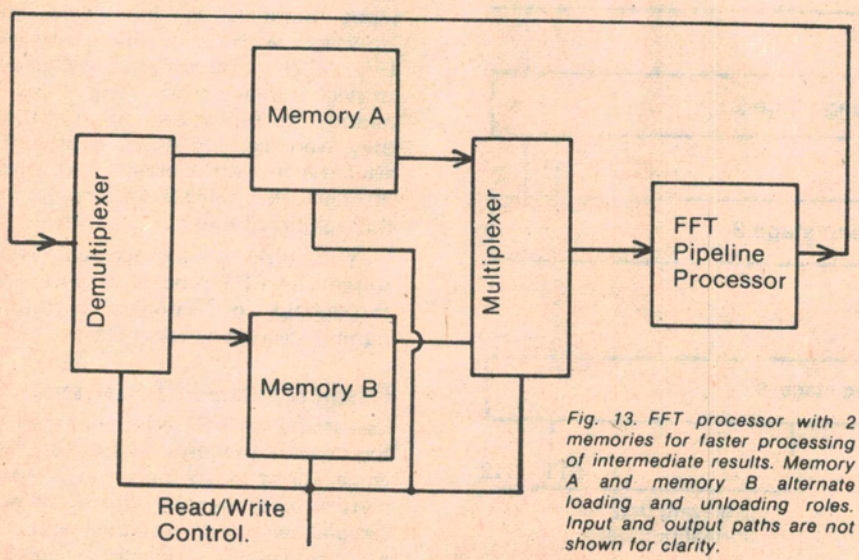


Fig. 13. FFT processor with 2 memories for faster processing of intermediate results. Memory A and memory B alternate loading and unloading roles. Input and output paths are not shown for clarity.