inmos

correlation and convolution with the IMS A100

10.1 Introduction

The correlation process is widely used in many electronic systems including instrumentation, communication, medical ultrasonics, radar, sonar, control systems and other signal processing environments. The basic reasons for this widespread use can be attributed to the many useful characteristics exhibited by the correlation process. These properties include

- The ability to recover a desired signal masked by noise or other interferences. This is particularly useful in noisy environments that arise in communication, radar, sonar and ultrasonic applications.
- Delay estimation capability which is essential in many applications including range measurement in navigation systems, radar, sonar and also system identification.
- The ability to recognize a given pattern within a signal.
- The auto-correlation of a signal is closely related to the power spectrum which has resulted in the application of the correlation process to spectral analysis.
- The correlation process provides a good characterization of many signals and has therefore been used in many prediction and estimation algorithms.

Convolution is closely related to the correlation process. Mathematically convolution is what happens in the process of filtering. It will be shown in the next section that both these functions involve a large number of multiplications and additions. Up to now, for the time domain implementation of these processes, many systems have used multiply-accumulator devices. Because of their inherent concurrency, the numerical evaluations involved in the convolution and correlation functions can be performed in parallel. But due to the high cost, power consumption requirement, and size restriction many digital systems use only a single (or possibly two) multiply-accumulator(s). This has resulted in a processing bottle-neck in the time domain evaluation of these functions. For example using a 16-bit multiply and accumulator chip available today it is possible, for a 32-point digital correlator, to achieve at best a sampling frequency of around 100 to 300KHz. This is further reduced as the number of correlation points increases. Additional complexities occur as some form of address generator has to be used to sequence the data and the reference coefficients through the multiply-accumulator chip.

The IMS A100 VLSI chip overcomes these problems by incorporating 32 multiply-accumulators on a single chip. The sampling speed of the IMS A100 ranges from 2.5 MHz to 10 MHz depending on the reference-waveform word-size. (4, 8, 12 or 16 bits). It is the true parallelism incorporated in the systolic structure of the IMS A100 that allows such speed increases. The architecture of the IMS A100 has been designed in such a way that large numbers of these chips can be cascaded to perform high precision correlations involving more than 32 points at full speed. Alternatively it is possible to use multidimensional index mapping to decompose a long correlation/convolution into a number of short ones which can then be carried out by usoing a single or a small number of devices.

By suitable allocation of the coefficients, the IMS A100 can be used to perform 3×3, 5×5, or larger two-dimensional image convolutions.

In this application note the concepts of correlation and convolution are first introduced followed by their IMS A100 implementation issues. Partitioning techniques for decomposing a long correlation/convolution into a number of short ones are then described. Next an example of a two-dimensional image convolution is given. Finally some application areas of correlation and convolution are summarised.

10.2 Correlation concepts

The correlation between two functions is a measure of their similarity. This is illustrated in figure 10.1 where three extreme cases are depicted. Figure 10.1a shows two waveforms which are absolutely identical and they have maximum positive correlation. The two waveforms in figure 10.1b are similar, except for their polarities and as such they have maximum negative correlation. Finally figure 10.1c shows one of the waveforms of figure 10.1a and a noise like signal. As these two waveforms are completely dissimilar the correlation between them is expected to be very small or even zero.

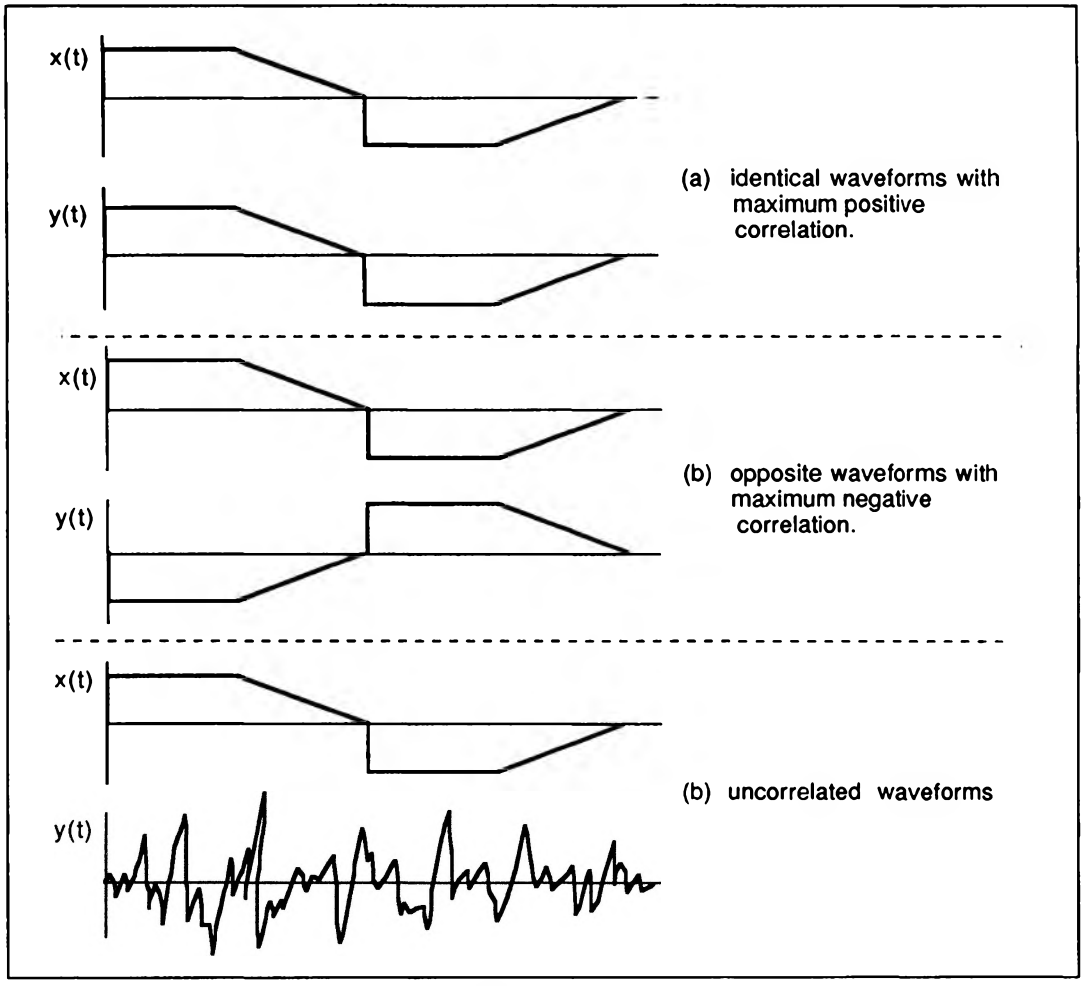


Figure 10.1 Illustration of correlation process

Mathematically the correlation function between two waveforms $x(t)$ and $y(t)$ is expressed as

$$R_{xy}(\tau) = \lim_{T\to\infty} \frac{1}{T} \int_{\frac{-T}{2}}^{\frac{+T}{2}} x(t)y(t+\tau)dt \tag{1}$$

$R_{xy}(\tau)$ is also referred to as cross-correlation between the two waveforms. For identical waveforms (ie correlating a waveform $x(t)$ with itself) the correlation function is denoted by $R_{xx}(\tau)$ and is called the auto-correlation function.

Equation (1) can be interpreted as follows:

The cross-correlation function, $R_{xy}(\tau)$ between the two waveforms $x(t)$ and $y(t)$ is obtained by shifting one of the two signals in time by an amount equal to τ (i.e. modifying $y(t)$ to $y(t+\tau)$), multiplying the shifted waveforms by the other signal and integrating the product.

If the waveforms are periodic with a period T_0, equation (1) can be modified to:

$$R_{xy}(\tau) = \frac{1}{T_0}\int_{\frac{-T_0}{2}}^{\frac{+T_0}{2}} x(t)y(t+\tau)dt \qquad (2a)$$

i.e. the integration is evaluated only over one period of the signal.

Figure 10.2 provides a graphical illustration of the process of cross correlation between two waveforms $x(t)$ (figure 10.2a) and $y(t)$ figure 10.2b. We start by multiplying the two waveforms and integrating over the interval $\frac{-T_0}{2} \leq t \leq \frac{+T_0}{2}$ yielding $R_{xy}(0)$. With $\tau = 0$, (ie no shift) it can be seen from figures 10.3a & 10.3b that there is no overlap between non-zero parts of the two waveforms resulting in $R_{xy}(0) = 0$ as shown in figure 10.2f. To evaluate $R_{xy}(\tau)$, the waveform $y(t)$ is left shifted by an amount equal to τ, giving $y(t+\tau)$, and the multiplication and integration is repeated. As the waveform $y(t)$ is shifted to the left, there will be no overlap between non-zero parts of $y(t+\tau)$ and $x(t)$ untill $\tau > \tau_1$, see figure 10.2c. At $\tau = \tau_1$, the non-zero parts of the two waveforms just begin to overlap. Figure 10.2d shows the position of $y(t)$ when it is shifted by $\tau = \tau_2$. Here the non-zero parts of $x(t)$ and $y(t+\tau_2)$ have have overlapped and the integration of the product of the two waveforms therefore yields a non-zero value for $R_{xy}(\tau_2)$ as shown in Figure 10.2f. As $y(t)$ is shifted further the non-zero overlapping section of the two waveforms and hence the value of $R_{xy}(\tau)$ increase. When $y(t)$ is shifted to the position shown in figure 10.2e, full overlap occurs and $R_{xy}(\tau)$ will attain its maximum value of $R_{xy}(\tau_3)$ as shown in figure 10.2f. Shifting $y(t)$ further causes the value of $R_{xy}(\tau)$ to decrease as the two waveforms pass each other. Figure 10.2f shows the complete cross-correlation function between the two waveforms. You can confirm the shape of this cross-correlation function by evaluating equation (2a).

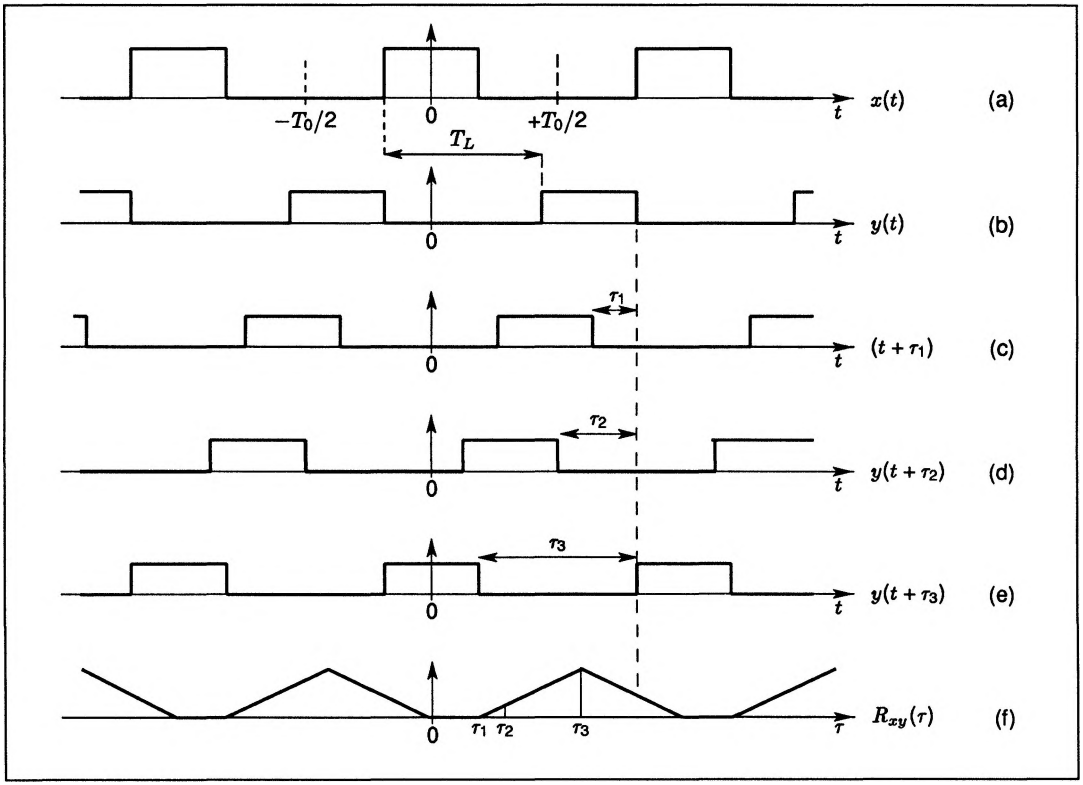


Figure 10.2 Graphical illustration of the correlation process

One interesting point to note here is that the maximum value of $R_{xy}(\tau)$ occurs at $t = \tau_3$ which is equal to the time-lag, T_L, between the two waveforms $x(t)$ and $y(t)$. This is how the correlation process is used to measure delays.

From figure 10.2 it can be seen that the cross correlation function could have been evaluated by shifting $x(t)$ to the right instead of left shifting $y(t)$. Mathematically this can be confirmed by defining a new variable $t' = t + \tau$ and substituting in (2a) which gives:

$$R_{xy}(\tau) = \frac{1}{T_0} \int_{\frac{-T_0}{2}}^{\frac{+T_0}{2}} x(t' - \tau) y(t) dt \tag{2b}$$

So far we have dealt with the correlation of analogue signals. For digital processing both waveforms $x(t)$ and $y(t)$ have to be sampled and digitalized. For discrete-time signals the process of correlation can be expressed as:

$$R_{xy}(mT) = \frac{1}{N} \sum_{k=0}^{N-1} x(kT) y((k+m)T) \tag{3a}$$

At time $t = kT$ equation (3a) requires future samples of $y(t)$. Similar to the analogue case, (equation 2b), the above equation can be modified so that it only uses past samples of $y(t)$, i.e.

$$R_{xy}(mT) = \frac{1}{N} \sum_{k=0}^{N-1} x((k-m)T) y(kT) \tag{3b}$$

In equation (3b), T, donates the sampling period and should be chosen to ensure that the sampling rate is greater than twice the signals bandwidth (Nyquist rate). For the sake of simplicity the factor, T, is usually dropped from the indices of equations (3a) & (3b), i.e.

$$R_{xy}(m) = \frac{1}{N} \sum_{k=0}^{N-1} x(k) y(k+m) \tag{4a}$$

and

$$R_{xy}(m) = \frac{1}{N} \sum_{k=0}^{N-1} x(k-m) y(k) \tag{4b}$$

Where k and m are used to index the samples and N is the number of correlation points involved. In practice the correlation size N will depend on the duration of the two functions, and on their periodicity if they are periodic.

From equations (4a) or (4b) it can be observed that direct evaluation of M samples of the cross-correlation function, R_{xy}, will involve $M \times N$ multiply-and-accumulate operations.

10.3 Convolution concepts

The convolution function is closely related to that of correlation. The convolution of two signals $x(t)$ and $y(t)$ is mathematically defined by:

$$C_{xy}(\tau) = \lim_{T \to \infty} \frac{1}{T} \int_{\frac{-T}{2}}^{\frac{+T}{2}} x(t) y(\tau - t) dt \tag{5}$$

This equation is very much similar to equation (1) defining the correlation process. Their difference is that in convolution the signal $y(t)$ is first time-reversed (i.e. is mirrored around $t = 0$) and then shifted by τ. This time-reversed and shifted signal is then multiplied by $x(t)$ and the product is integrated over all t's. Figure 10.3 graphically illustrates the process of convolution.

The process of convolution occurs in filters where the output of a filter is in fact the convolution of the input function, $d(t)$, and the impulse response, $h(t)$, of the filter (see the application note entitled 'Digital Filtering with the IMS A100'):

$$f(\tau) = \int_{-\infty}^{+\infty} d(t) h(\tau - t) dt \tag{6}$$

where $f(\tau)$ is the filter output.

For discrete-time signals equation (5) becomes

$$C_{xy}(m) = \frac{1}{N} \sum_{k=0}^{N-1} x(k) y(m-k) \tag{7}$$

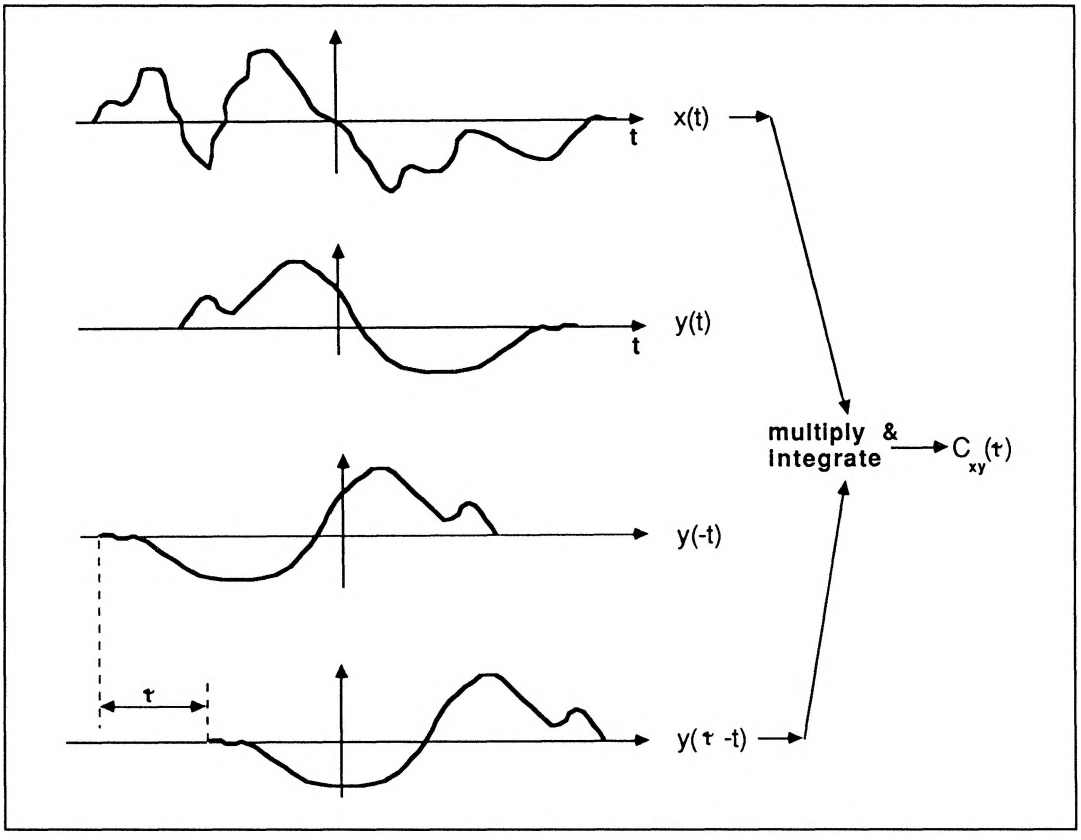


Figure 10.3 Illustrating the convolution process

which defines the convolution function for digital signals. Notice that like correlation, convolution involves carrying out N multiply-and-accumulations for each sample of $C_{xy}(m)$. Due to the high degree of similarity between correlation and convolution functions, the same hardware can be used to perform both functions. All that needs to be done is to time-reverse one of the waveforms for the convolution process.

The following two sections deal with hardware implementations for the correlation and convolution functions. The first section deals with the conventional approach involving multiply-and-accumulator chips and points out the processing bottle-necks associated with these solutions. The second section shows how the IMS A100 signal processor can be configured to perform these functions efficiently and simply, at speeds not economically feasible with the conventional approach.

10.4 Conventional hardware for time-domain evaluation of correlation

As discussed earlier, the processes of correlation and convolution are based on multiplying a delayed version of a sequence of samples by another sequence and summing the products. Conceptually this could be mechanised, as shown in figure 10.4, by providing two shift registers to hold all the values of x's and y's required for the computation, a further shift register to provide the delay (mT), an array of multipliers for forming the products, and a multi-input adder for the final summation. In the example of figure 10.4 the output would correspond to $R_{xy}(2)$, as a delay of two stages is incorporated in the path of signal $x(kT)$ giving $x((k-2)T)$ (see equation 3b).

Up to now, due to the large number of multipliers and adders involved, it has not been possible to eco-

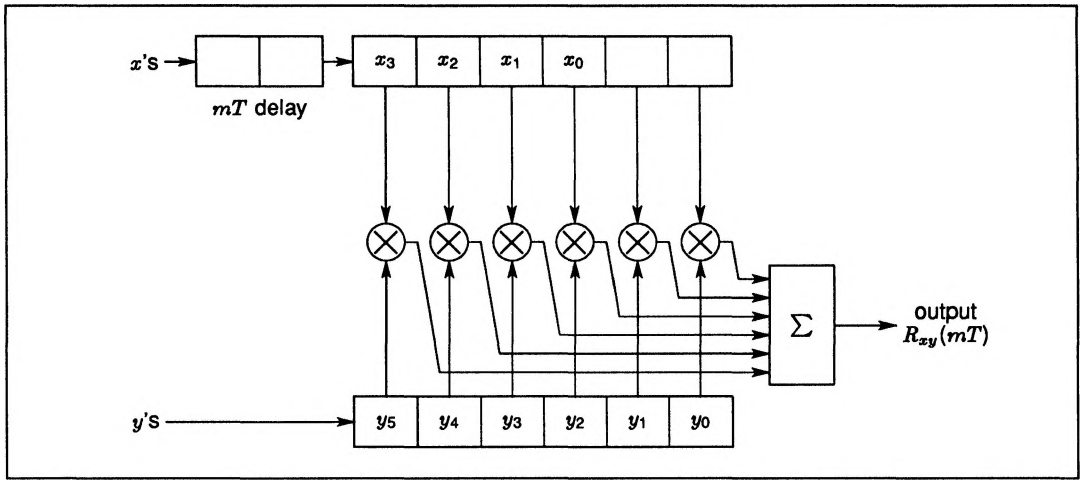


Figure 10.4 Schematic diagram for an ideal correlator hardware

nomically implement high precision correlators directly in the form given by figure 10.4. Instead to minimize the size, cost and power consumption, a single multiply-and-accumulator is usually used and time-shared between all the multiplications. Figure 10.5 shows a schematic block diagram of a conventional correlator implementation. The system consists of memories to hold samples of the two signals to be correlated, a multiply-accumulator and an address generator hardware which is responsible for sequencing the correct order of signal samples through the multiply-and-accumulator. The obvious disadvantage of this arrangement is the processing bottle-neck caused by using a single multiply-and-accumulator to sequentially evaluate what is inherently a concurrent problem. Assuming a multiply-accumulate time of T_{mac}, for an N-point correlator implemented using a single multiply-accumulator, the maximum sampling rate would be

$$f_{max} = \frac{1}{NT_{mac}} \tag{8}$$

For example if T_{mac} = 100ns and N = 100, then a signal sampling frequency of at most 100kHz would be possible.

Many applications such as radar and communication require faster processing rates than can be achieved using a single multiply-and-accumulator. (Some improvements can be achieved by carrying out the processing in the frequency domain at the cost of introducing some complexity. However here we are only concerned with the time-domain approach. A separate application note entitled 'Discrete Fourier Transform with the IMS A100' deals with the time-domain to frequency-domain transformations)

In applications where a fast processing rate is essential, a trade-off is often made between the correlator precision and its speed. For example if one or both of the signals x and y are assumed binary, the multiplications become simple binary AND operations, and it would be possible to implement a high speed low precision correlator. In fact many correlator chips available today are of this type and have very low precision compared to those implemented from multiply-accumulators.

The IMS A100 chip on the other hand is the first high-precision high-speed VLSI implementation of a single-chip correlator. It provides a numerical accuracy in excess of that of the 16-bit multiply and accumulators while allowing sampling rates in the MHz region. The next section illustrates how this chip can be used to perform fast and highly accurate correlation and convolution functions.

10.5 The IMS A100 implementation of correlation/convolution

The IMS A100 is a 32-stage correlator (convolver) in which the samples of the two signals to be correlated can be represented as up to 16-bit words. This corresponds to a signal dynamic range of 96 dB's. A number

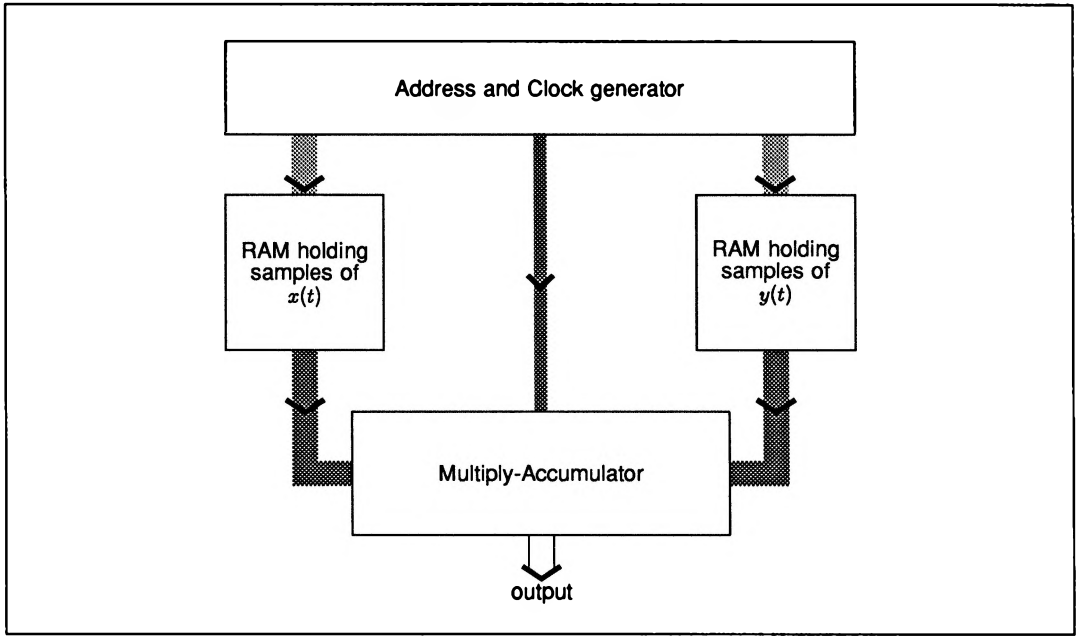


Figure 10.5 Block diagram of a conventional correlator/convolver

of these devices can also be cascaded, without the need for any external components, to provide much longer correlators while preserving a high degree of accuracy. The IMS A100 chip (or cascaded chips) can be fully memory mapped and used as a peripheral accelerator to a host processor.

To understand the architecture of the IMS A100 let us first consider the basic function of a simple 3-point correlator shown in figure 10.6a. The three samples of the first signal (reference signal) i.e. x_0, x_1 and x_2 are loaded in three registers feeding an array of three multipliers. The samples of the second signal i.e. y_0, y_1, y_2,.... are fed into a 3-stage shift register whose outputs are also connected to the multipliers. A three input adder combines the products to give the correlation function. As the samples of the signal y are shifted through the shift register, the output of this hypothetical correlator (assuming the shift register is reset at the start) will be:

$$y_0x_2,\ y_0x_1 + y_1x_2,\ y_0x_0 + y_1x_1 + y_2x_2,\ y_1x_0 + y_2x_1 + y_3x_2, \ldots\ldots$$

The correlator structure in figure 10.6a can be modified to that given in figure 10.6b without affecting the functionality. In figure 10.6b the multi-input summation process is avoided and replaced by a chain of delay-and-add units. The input, supplying the signal y, is also made common to all of the multipliers. Note also that the signal samples x_0, x_1, x_2 are stored in the opposite direction to that of figure 10.6a. Supplying the input sequence of samples $y_0, y_1, y_2, y_3, \ldots\ldots$ to the structure of figure 10.6b and simultaneously shifting the partial products along the delay-and-add chain, it is straightforward to confirm that the output sequence would be

$$y_0x_2,\ y_0x_1 + y_1x_2,\ y_0x_0 + y_1x_1 + y_2x_2,\ y_1x_0 + y_2x_1 + y_3x_2, \ldots\ldots$$

This sequence is absolutely identical to that obtained from figure 10.6a. In other words the structure in figure 10.6a & b have identical functionality and both can be used to perform correlation between two sequences. The IMS A100 architecture is based around this modified structure. The major processing part of the chip incorporates 32 multipliers and a 32-stage delay-and-add chain as shown in figure 10.7.

At this point the interested reader is advised to consult the data sheet of the IMS A100 for full details.

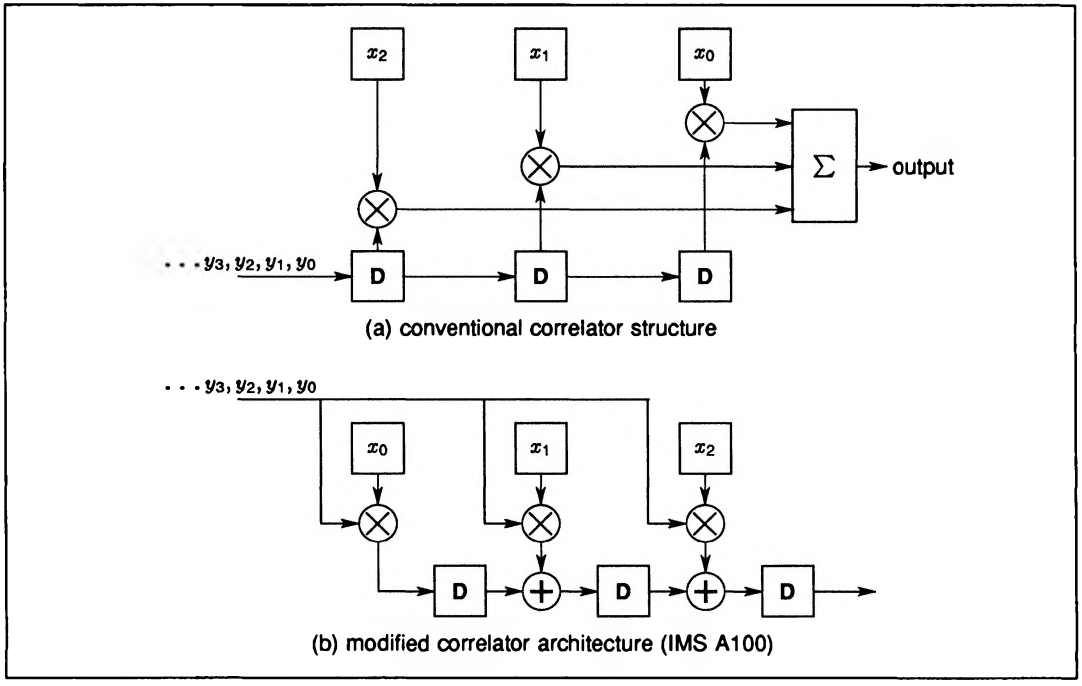


Figure 10.6 Relating the IMS A100 architecture to that of a correlator

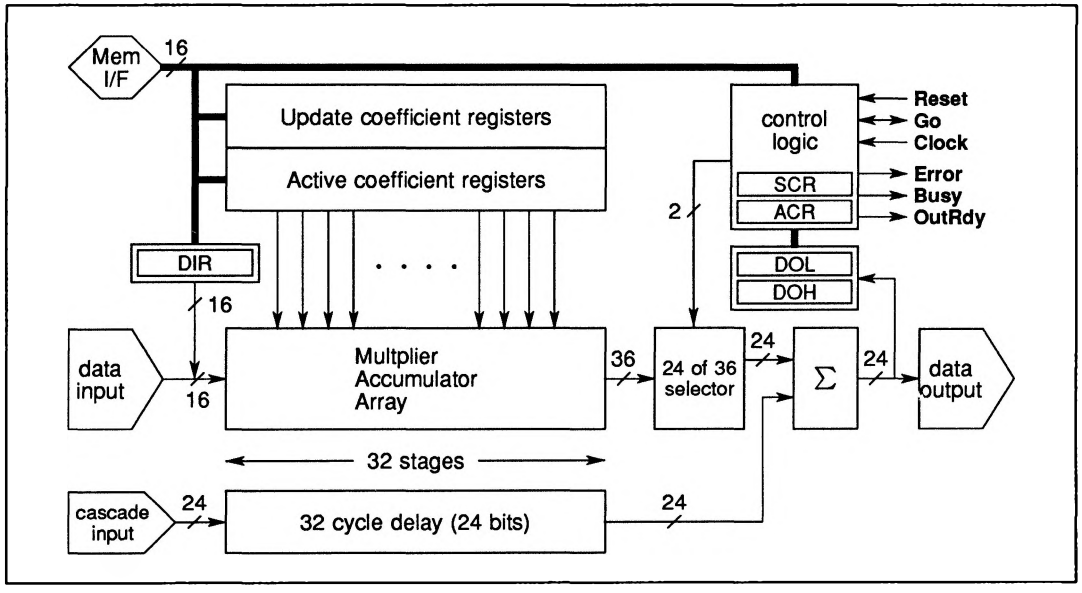


Figure 10.7 User's model of the IMS A100

In order to correlate two sequences $x(k)$ and $y(n)$, the samples of one of the two signals, say $x(k)$'s, should be stored in one set of the IMS A100's coefficient registers. These samples should be loaded from left to right with the last sample of $x(k)$ stored in the coefficient register associated with the last multiplier. If the reference waveforms $x(k)$ is less than 32-samples long, any unused left-most coefficient registers should be set to zero. For a 30-sample reference signal, this allocation is depicted in figure 10.8.

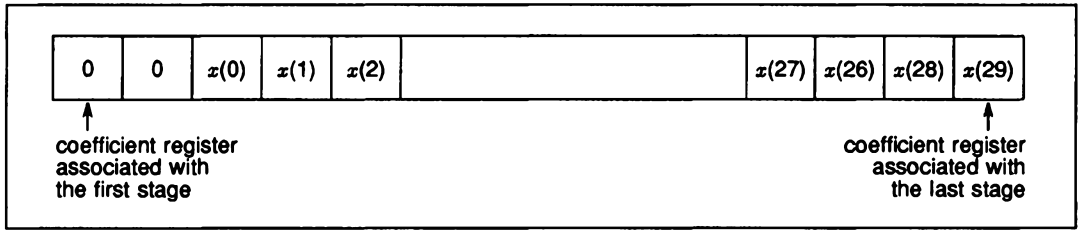


Figure 10.8 Example of the reference signal allocation for a 30 point correlator

The samples of the other signal $y(n)$ are then applied to the input of the IMS A100. As shown earlier the output sequence would correspond to the cross-correlation function of the two signals. If the two signals $x(k)$ and $y(n)$ are to be convolved rather than correlated, the reference signal $x(k)$ should be loaded in the coefficient registers in the opposite direction. The register allocation for a 30-point convolver is shown in figure 10.9. As discussed in the data sheet, the IMS A100 processor has two sets of coefficient registers. At any instant in time one set of coefficients is applied to the multiplier array, whilst the other set can be accessed via the IMS A100 memory interface. For correlations (convolutions) dealing with real signals, one set of these coefficients would be sufficient. The second set can be used to hold a different reference signal and if necessary the function of the two memory banks can be interchanged by performing a write operation to the 'Bank swap' bit of a control register. Such an operation would initiate the correlation (convolution) of the input signal with the second reference waveform. The existance of the two coefficient register sets and the continuous bank-swap mode allows the IMS A100 to perform complex (correlation)convolution, where both the reference and the input signal have real and imaginary components. This configuration is discussed in the application note 'Complex Processing with the IMS A100'.

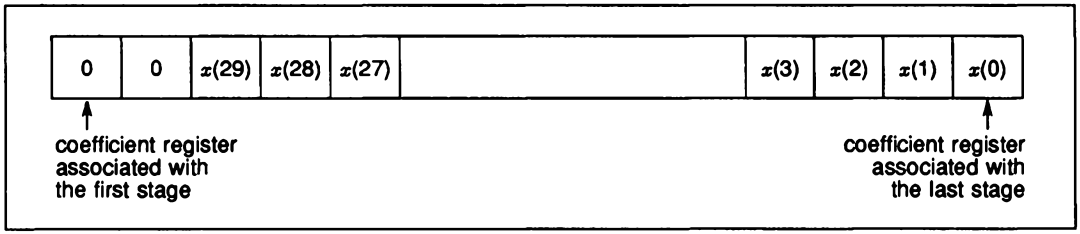


Figure 10.9 Coefficient register allocation for a 30 point convolution

For the IMS A100 the data-word length is 16 bits whilst the coefficient-word length can be programmed to be 4, 8, 12 or 16 bits. The maximum data throughput (the sampling rate) is a function of the coefficient size. Table 10.1 relates the coefficient size to the maximum sampling rate and indicates the effective number of multiply-and-accumulate operations per second in each case. The last column shows the effective number of multiply-and-accumulates when four devices are cascaded.

The strength of the IMS A100 can be appreciated by comparing the effective number of multiply-and-accumulations/sec with that of multiply-accumulator IC's which range from 5–10 million/sec.

As discussed in the A100 data sheet, in order to preserve complete numerical accuracy no truncation or rounding is carried out on the partial products in the multiply-and-accumulation array. The output is thus calculated to 36 bit precision which ensures no overflows. A barrel-shifter at the output of the multiply-and-accumulate array allows 24 bits from these 36 bits to be selected (sign-extended if necessary) and rounded for output. This selection can be programmed via a control register. The programming details can be found in the IMS A100 data sheet.

Coefficient word size (bits)	Sampling rate (MHz)	Effective number of multiply-and-accumulates (Millions/second)	
		Single A100	Four A100s
4	10	320	1280
8	5	160	640
12	3.3	106	424
16	2.5	80	320

Table 10.1

The architecture of the IMS A100 has been designed to allow large numbers of these devices to be cascaded for correlations (convolutions) involving more than 32-stages, the devices can be cascaded while preserving a high degree of accuracy and without the need for any external components. This is made possible by incorporating on chip a 32 stage, 24-bit wide, shift register and a 24-bit adder which combines the output of the barrel-shifter with that of the 32-stage shift register (see figure 10.7).

The IMS A100 chips can be cascaded by simply connecting the output of the each device to the cascade input of the following device. The input is common to all cascaded devices. The effect of such an arrangement is that the output of the first device is delayed by 32 cycles, before being added to that of the next device. Figure 10.10 illustrates how, for example, a 64-point correlator can be implemented using two IMS A100 devices. The allocation of the reference signal samples is also indicated in this illustration. In this arrangement the barrel-shifter in each device acts as a data scalar (with rounding). The cascading process can be considered as a block-floating point operation where the common exponent is determined by the extent of the shift carried out by the barrel-shifter. With this cascading technique a very high degree of accuracy is preserved because the output scaling is only performed after every 32-multiply and accumulation stages and not at any intermediate stage.

For convolution purposes the reference signal should be loaded into the coefficient stage in the opposite direction to that shown in figure 10.10.

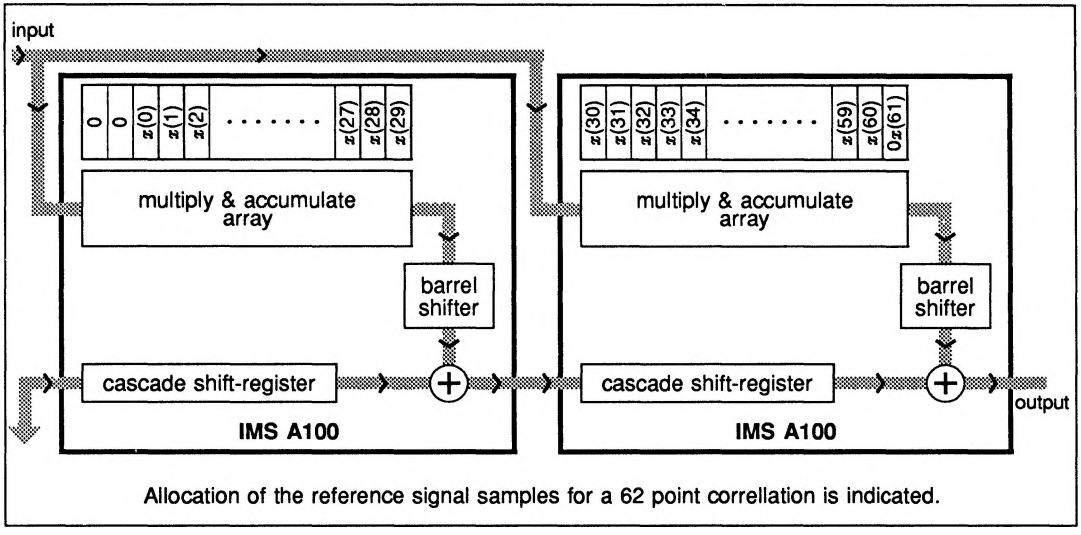


Figure 10.10 Cascading two IMS A100 devices to obtain a 64 point correlator

A very important feature of the IMS A100 transversal filter is that the part is fully memory mappable. Apart from the two coefficient memory banks, which can be accessed via the IMS A100 standard memory interface, the input and output of the device are also accessible from the same interface. This feature allows the part

to be used either with its input and output data communicated through the dedicated ports or through the memory interface. The latter options allows the device to be easily interfaced to a host processor and used as a high speed peripheral. The status and control registers of the IMS A100, accessible via the memory interface, provide full control of the part by the host processor.The memory interface can also be used as a facility for system diagnostics, as the host processor can act as a watch-dog in systems involving arrays of IMS A100's. Full specification of the IMS A100, its status and control registers and its standard memory interface are detailed in the data sheet available from INMOS.

10.6 Decomposition of long correlations and convolutions

A single IMS A100 device is effectively a 32-tap correlator (convolver) in which the samples of the two signals to be correlated can be expressed in upto 16-bit words. As described earlier, one method to deal with correlations/convolutions involving more than 32 points is to use several cascaded devices to achieve a longer correlator/convolver. For such an arrangement, and with 16-bit coefficients, tha data rate can be as high as 2.5 Million samples/sec.

Alternatively it is possible to use various decomposition techniques to partition a long correlation/convolution into a number of shorter ones, which can then be carried out by a single or a small number of IMS A100 devices. The host machine would merely combine the results from these short correlations/convolutions to obtain the overall result. The advantage of this approach, compared to using single MAC based processors, is a significant reduction in the required memory bandwidth. This is why even a medium-speed general purpose microprocessor can achieve a very high performance when combined with the IMS A100.

A simple way to decompose a long correlation/convolution of length N, between waveforms x and y, is to break up one of the waveforms, say x, into consecutive blocks of 32 sample. Each one of these blocks can then be correlated/convolved with the whole of the waveform y by loading each block into the IMS A100 coefficient registers, and using y as the input sequence. The output from these correlations/convolutions can then be combined by displacing each partial result by 32 samples, with respect to the previous one, and performing an addition operation. Note that the coefficient registers, containing blocks of waveform x, need only be updated once every time the whole of the waveform y is fed through the device, resulting in a significant saving in the memory bandwidth. The block size of 32, suggested above, whould mean that a single IMS A100 would be sufficient. However processing speed can be improved by using cascading devices to perform these partial correlations/convolutions. With suitable memory mappings, hosts such as INMOS transputers can use their on-chip DMA engine to feed the IMS A100 devices with the samples of the waveform y.

A more complicated decomposition technique, to be described here, is based on the multidimensional index mappings (references 1 & 2). These techniques are applicable to cyclic convolutions/correlations. However all convolutions/correlations can be made cyclic by adding zero terms to the end of the data blocks. As an example, consider the following cyclic correlation:

$$C(k) = \sum_{n=0}^{N-1} x(k+n)y(n) \tag{9}$$

where the indices are evaluated modulo N. The arrays C, x, and y can be mapped into multidimensional arrays C', x', and y', the requirement being that the mapping should be one-to-one and cyclic in at least one dimension. The map, in general, can assume many different forms, but the one particularly useful is the linear form. For a simple two-dimensional decomposition such a map would be of the form:

$$n = (M_1 n_1 + M_2 n_2) \ mod \ N. \tag{10}$$

Note that n is evaluated modulo N, making the map cyclic in n. In order for this map to be unique and one-to-one, the mapping constants M_1 and M_2 must satisfy certain conditions. These conditions are summarised in section 6 of the IMS A100 Application Note 2 which is available from INMOS and will not be repeated here.

As an example let us map the arrays in equation (9) into two-dimensional matrices of dimensions N_1 and N_2 where $N = N_1 \times N_2$, we can use the mapping given by equations (10) for n and k giving

$$C(M_1k_1 + M_2k_2) = \sum_{n_1=0}^{N_1-1} \sum_{n_2}^{N_2-1} x(M_1k_1 + M_2k_2 + M_1n_1 + M_2n_2)y(M_1n_1 + M_2n_2) \tag{11}$$

or

$$C'(k_1, k_2) = \sum_{n_1}^{N_1} \sum_{n_2}^{N_2} x'(k_1 + n_1, k_2 + n_2)y'(n_1, n_2) \tag{12}$$

This is now a true two-dimensional convolution which can be made cyclic along n_1 if M_1 is made a multiple of N_2, and/or cyclic along n_2 if M_2 is made a multiple of N_1. With these conditions, inspection of equation (12) shows that the long N-point circular correlation can be performed by N_1^2, N_2-point correlations or N_2^2, N_1-point correlations. This involves correlating each row (or column) of the matrix y' with all the rows (or columns) of the matrix x'. These short circular correlations can be efficiently performed by the IMS A100, with the host merely adding partial results. The approach is particularly efficient as it is possible to load one row (or column) of the matrix y' into the coefficient memory of the device and to feed all the rows (or columns) of the matrix x' successively to the input of the device to obtain partial results, for the elements in the matrix C'. The fact that with this algorithm, the coefficient memories need only be updated occasionally (once every time all the elements of the matrix x' are fed into the device) results in an impressive reduction in the memory bandwidth requirement. This is why, even with a general purpose microprocessor, as the host, very impressive perfomance can be achieved.

In the example given here, we concentrated around a two-dimensional mapping. It is important to realise that the same decomposition concepts can be extended to more dimensions. The easiest way to see this is to start with a two dimensional decomposition and then partition the rows of the two-dimensional matrices further. For example if

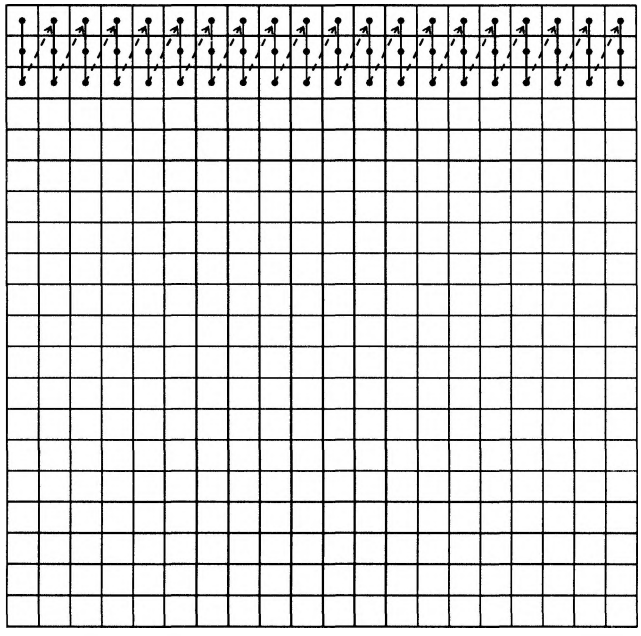
$$N = N_1 \times N_2 \times N_3$$

the original N-point correlation can be carried out via N_3^2, $N_1 \times N_2$-point correlations. However, each one of the $N_1 \times N_2$-point correlations can further be decomposed, as before into N_1^2, N_2-point correlations.

10.7 2-D image convolutions with the IMS A100

Many applications including image processing require 2-D convolutions and correlations. Such operations are needed in image filtering, edge detection, etc. There are many ways that the IMS A100 can be used to speed up these operations. This section gives an example of how the device can be used to perform 3×3, 5×5, or larger convolutions.

Figure 10.11a shows a 20×20-pixel image which is to be convolved with the 3×3 reference matrix given by figure 10.11b. One way to achieve this is to load the reference matrix, as shown in figure 10.11c, in one of the IMS A100 coefficient register banks, and sequence the image data through the device as shown by the arrowed path in figure 10.11a. In this way every third output sample of the IMS A100 would correspond to a valid filtered pixel for the second row of the image. To proceed, the same sequence, moved down by one row, is then passed through the device which provides the filtered results for the nest row and so on. A single IMS A100 can deal with reference matrices as big as 5×5.



(a) 20×20 pixel image, arrows show the required data sequencing

a	d	g
b	e	h
c	f	i

(b) 3×3 convolution matrix

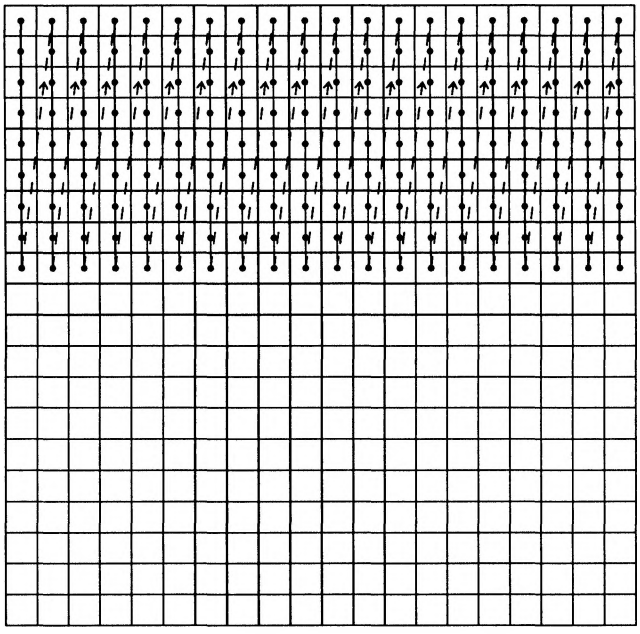
0	0	0	0	.....		0	0	a	b	c	d	e	f	g	h	i

(c) Coefficient register allocation for the 3×3 convolution

Figure 10.11 Example of a 3×3 image convolution/correlation with the IMS A100

An alternative arrangement which gives a better throughput is one where, as shown in figure 10.12a, 7 zeroes are inserted in the IMS A100 coefficient registers (between terms corresponding to the columns of the reference matrix). The data sequencing would be as shown in figure 10.12c, where ten pixels from a given column are fed through the device before moving to the next column. In this scheme the first nine rows of the image are filtered in one scan, with 80% of the output data samples being valid. (Note that, using a single device, the number of inserted zeroes can be increased from 7 to 11, allowing 13 image rows to be filtered in each scan.)

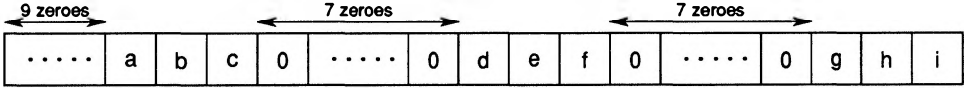
The examples given here are just a small subset of possible arrangements. Remembering that the IMS A100 devices can be cascaded or used in parallel, numerous other implementations for image processing become possible.



(a) 20×20 pixel image, arrows show the required data sequencing

a	d	g
b	e	h
c	f	i

(b) 3×3 convolution matrix



(c) Coefficient register allocation for the 3×3 convolution

Figure 10.12 Improved version of the 3×3 image convolution/correlation with the IMS A100

10.8 Some application examples of correlation and convolution

Correlation and convolution are encountered in numerous applications of digital signal processing, this section summarises some of the application areas where these techniques are used.

10.8.1 Delay and periodicity estimation

The correlation process can be used to estimate the time delay between two similar signals. Figure 10.13 shows two signals $x(t)$ and $y(t)$ which are identical in shape but have a time delay between them. If these two signals are correlated the cross-correlation function would attain a maximum when $y(t)$ is delayed by an amount equal to the delay between the two waveforms. This is illustrated in figure 10.13c where the peak of the cross-correlation function occurs at $t = T_d$ where T_d is the delay between the two waveforms. This technique has applications in areas such as radar, sonar and medical ultrasonics where a measurement of the time delay between the transmitted signal and the return echo from an object gives an indication of the range of that object.

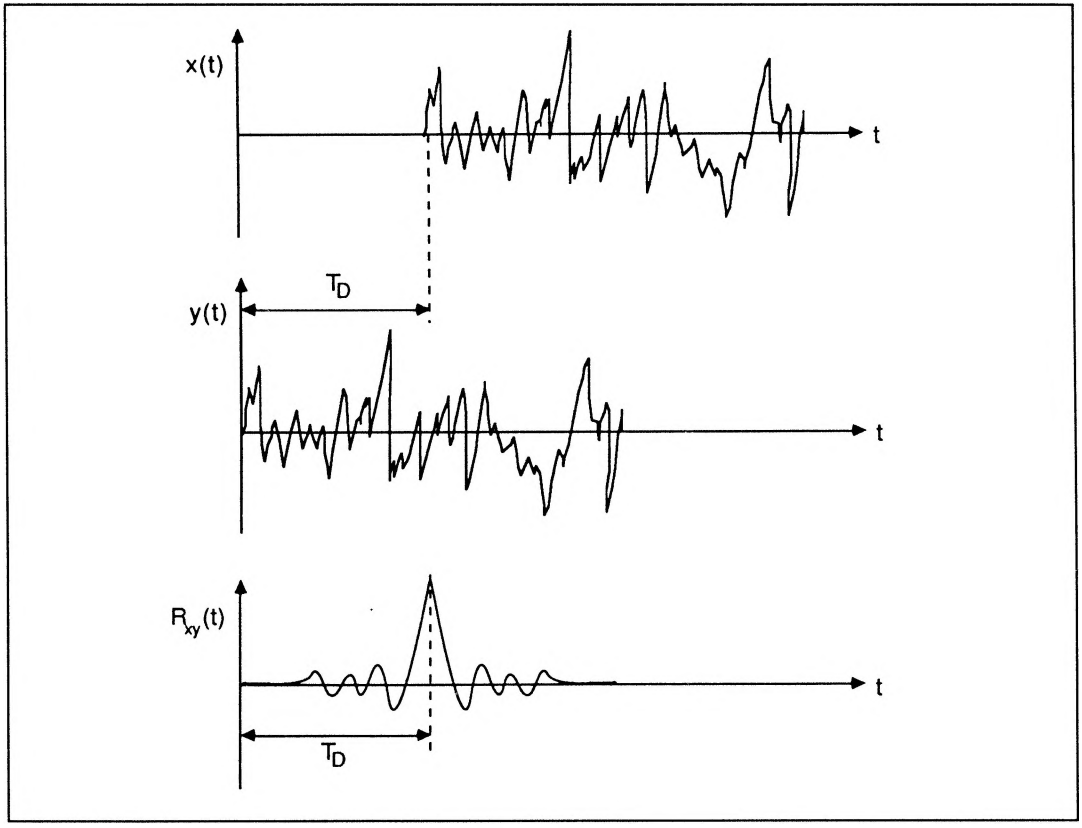


Figure 10.13 Delay estimation using correlation process

The same technique can also be used to measure the period of a repetitive signal. This can be achieved by correlating the signal with itself i.e. by calculating its auto-correlation function as illustrated in figure 10.14 the auto-correlation of a periodic signal exhibits peaks, spaced a distance, T_0, apart where T_0 is the period of the signal. One application of this technique is pitch-period measurement in speech signals. The time gap between the peaks in the auto-correlation function of a segment of speech provides an estimate for the pitch period of voiced speech.

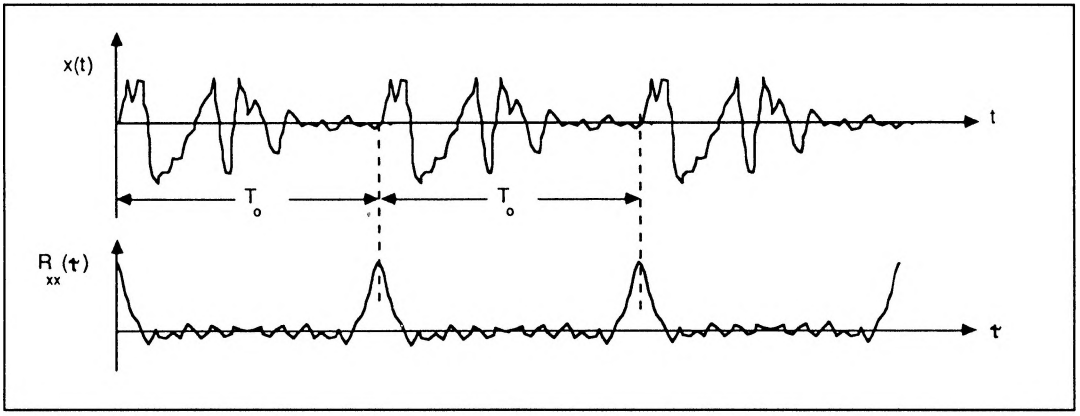


Figure 10.14 Application of correlation to the periodicity measurement

10.8.2 Noise reduction using correlation techniques

In many real-world applications the signals to be processed are immersed and possibly masked by noise. Such situations occur in noisy communication channels, long-range radar and sonar systems. In such cases correlation techniques can be used to extract and detect the signal from the background additive noise. This is achieved by correlating the noisy signal with a replica of the expected signal waveform. While the noise is uncorrelated with the replica signal, the signal immersed in noise will strongly correlate with the replica signal giving a large output value, well above the background noise. Mathematically this can be argued as follows (the proof here is not rigorous but does make the point):

Let the signal waveform consist of N samples values $s_0, s_1, s_2, \ldots\ldots s_{N-1}$. Suppose this signal is correlated by samples of a white noise having a standard deviation of σ_n (and variance σ_n^2). The ratio of the signal power to that of the noise prior to any processing is thus:

$$\frac{Signal Power}{Noise Power} = \frac{\sigma_s^2}{\sigma_n^2}$$

where σ_s^2 is the signal variance. Suppose we correlate this noisy signal with a replica of the original signal in an N-point correlator. At the instant when the signal waveform masked by the background noise is aligned with its replica in the correlator, the output attains its maximum. At this instant the amplitude of signal component at the output of the correlator would be

$$S_{out} = s_0^2 + s_1^2 + s_2^2 + s_3^2 + \ldots\ldots + s_{N-1}^2 \simeq N\sigma_s^2 \tag{13}$$

The corresponding output signal power would thus be:

$$Output\ Signal\ Power = N^2\sigma_s^4. \tag{14}$$

The noise would also be modified by the operation of the correlator. In this case each output noise sample is equal to the sum of weighted input noise samples-the weighting coefficients being, of course, the samples of the reference signal. Hence each output noise sample is equal to the summation of N independent random numbers having standard deviations $s_0\sigma_n, s_1\sigma_n, s_2\sigma_n, \ldots\ldots, s_{N-1}\sigma_n$. Since variances are additive in this case, the variance of the output noise samples is therefore

$$\sigma_{n_{OUT}}^2 = s_0^2\sigma^2 + s_1^2\sigma^2 + \ldots\ldots s_{N-1}^2\sigma^2 = N\sigma_n^2\sigma_s^2 \tag{15}$$

The ratio of the output signal power to that of the output noise is thus

$$\left(\frac{S}{N}\right)_{OUT} = \frac{Output\ Signal\ Power}{\sigma_{n_{OUT}}^2} = \frac{N^2\sigma_s^4}{N\sigma_n^2\sigma_s^2} = N\frac{\sigma_s^2}{\sigma_n^2} = N\left(\frac{S}{N}\right)_{INPUT} \tag{16}$$

This indicates that the correlation process improves the signal to noise ratio by

$$C_G = 10 \log_{10}(N) \quad dB's \tag{17}$$

which is defined as the 'Correlator Gain'.

10.8.3 Pulse-compression

Another application of correlation is in radar and sonar systems where pulse compression techniques are used to improve range resolution of the systems. In many active sonar and pulsed radar systems, a short pulse is transmitted followed by a listening period that represents a 'look' in the range dimension. The two way propagation duration, i.e. the time it takes for the pulse to travel to a target and back gives an indication of the range of that target. The range resolution i.e. the shortest distance between two targets that can be resolved, is equal to $\frac{\tau c}{2}$ where τ is the pulse duration and c is the speed of wave propagation in the medium. For example for a radar system a $10\mu s$ pulse corresponds to a range resolution of 1.5km. Better range resolutions necessitate a shorter pulse. Unfortunately the transmitted pulses cannot be made too short. This is because most systems are peak-power limited and a shorter pulse means less signal power which in turn can severely limit operational range of the system.

Pulse compression techniques allow a radar or sonar to utilize a long pulse to achieve large radiated energy, but simultaneously to obtain the range resolution of a short pulse. This is accomplished by using a coded signal instead of a simple CW pulse. At the receiver the returned signal is correlated with a replica of the coded transmit signal. The returned signal would only correlate heavily with the replica for a short time, corresponding to when the echoes are aligned with the replica. This results in a narrow pulse appearing at the output of the correlator, everytime a match occurs. A signal that is commonly used in pulse-compression techniques is the lineal FM signal. An example of such a signal is depicted in figure 10.15a. The autocorrelation of such a waveform is shown in figure 10.15b. Note that the autocorrelation function has a narrow peak at the origin, with small side lobes elsewhere, i.e. the initially long FM pulse is 'compressed' into a narrow pulse after the autocorrelation process.

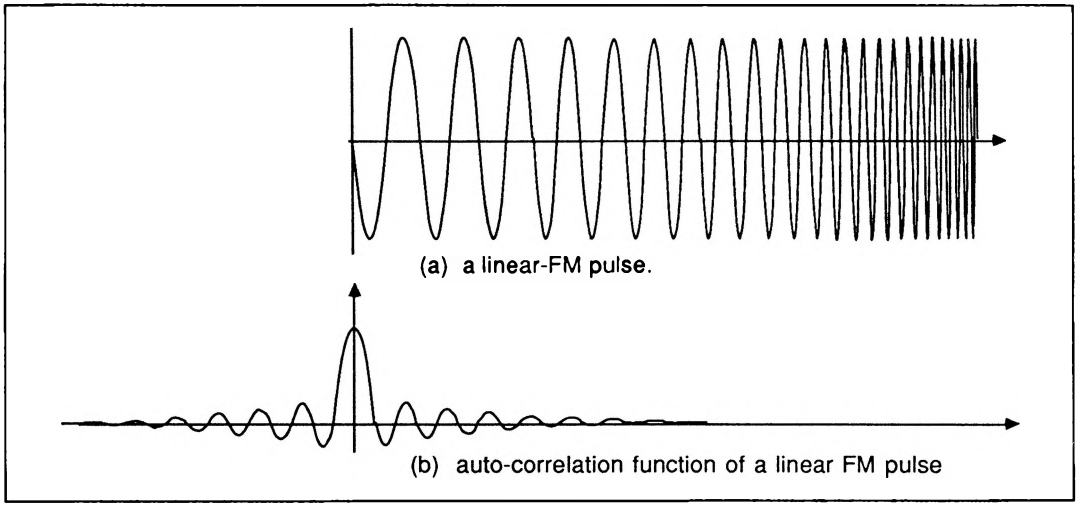


Figure 10.15 Pulse compression using a coded signal

It can be shown that the degree of compression is equal to BT where B is the bandwidth of the coded pulse and T is its duration. The effective pulse duration, as far as the range resolution is concerned, will thus be:

$$Effective\ Pulse\ Duration = \frac{T}{BT} = \frac{1}{B} \tag{18}$$

If the $10\mu s$ pulse in the previous example is coded in such a way that its bandwidth becomes 5MHz, the effective pulse duration would be:

$$\frac{1}{5 \times 10^6} = 0.2\mu s$$

This corresponds to a range resolution of 30 metres.

10.8.4 System identification using correlation

Another important application of cross-correlation is in the use of random-noise test signals to identify the impulse response of a system. For a system with an unknown impulse response $h(t)$, the output $y(t)$ is related to an input $x(t)$ by

$$y(t) = \int_{-\infty}^{+\infty} h(u)x(t-u)dt \tag{19}$$

The cross-correlation between the input $x(t)$ and the output $y(t)$ is defined by

$$\begin{aligned} \Phi_{xy}(\tau) &= \lim_{T\to\infty} \frac{1}{T} \int_0^T x(t)y(t+\tau)dt \\ &= \lim_{T\to\infty} \frac{1}{T} \int_0^T x(t) \int_{-\infty}^{+\infty} h(u)x(t+\tau-u)dudt \end{aligned} \tag{20}$$

Using simple mathematical manipulation it can be shown that

$$\Phi_{xy}(\tau) = \int_{-\infty}^{+\infty} h(u)\Phi_{xx}(\tau-u)du \tag{21}$$

i.e. The cross-correlation between $x(t)$ and $y(t)$ is the convolution of the impulse response $h(t)$ with the auto-correlation of the input signal.

If the input signal consists of broad-band white noise then its auto-correlation function, $\Phi_{xx}(\tau)$, would be an impulse (since a noise signal only correlates with itself at zero delay, $\tau = 0$). Referring to equation (21) it therefore follows that for broad-band noise input, the output $\Phi_{xy}(\tau)$ would be a direct measure of $h(\tau)$ since

$$\Phi_{xy}(\tau) = \int_{-\infty}^{+\infty} h(u)\delta(\tau-u)du = h(\tau) \tag{22}$$

Figure 10.16 illustrates the technique.

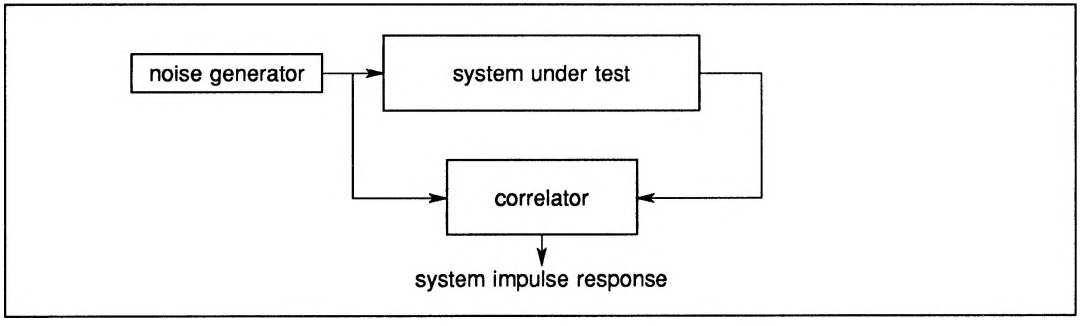


Figure 10.16 System identification using correlation

10.8.5 The Discrete Fourier Transform (DFT)

DFT has application in many signal processing areas, including speech processing, radar, sonar, image processing and control. This transform has often been performed using Cooley and Tukey radix-2 FFT algorithm (reference 3). This algorithm reduces the number of multiplications compared to direct evaluation of the DFT at the expense of complicating the required indexing.

Other algorithms have also been developed which allow the evaluation of the DFT via correlation (convolution) techniques (references 1, 2, 4, & 5). The IMS A100 device can be used to perform high speed DFT's based on these convolutional algorithms. Using the IMS A100 as a peripheral to a general-purpose microprocessor converts a slow host into a high-performance DFT processor. A separate application note available from INMOS describes how these algorithms can be implemented using the IMS A100 devices.

10.9 References

1 Burrus C.S., 'Index mappings for multidimensional formulation of the DFT an convolution', *IEEE Trans. on ASSP*, Vol.25, June 1977, pp 239-242.

2 Burrus C.S., Parks T.W., 'DFT/FFT and convolutional algorithms-theory and implementation', Wiley-interscience Publication, New York 1985.

3 Cooley J.W., Tukey J.W. 'An algorithm for the machine calculation of complex Fourier series', *Math. Comput.*, Vol.19, pp 297-301, April 1965.

4 Rader C.M., 'Discrete Fourier transforms when the number of data samples is prime', *Proc. IEEE*, Vol.56, pp 1107-1109, June 1968.

5 Rabiner L.R., *et al*, 'The chirp z-transform algorithm and its application', The Bell System Technical Journal, May-June 1969, pp 1249-1292.