NON-VOLATILE STORAGE IN PORTABLE MEASUREMENT SYSTEMS:

APPLICATION TO A CLASS OF MEASUREMENT PROBLEMS

R.F. KNIGHT

THE CITY UNIVERSITY
LONDON
1984

ABSTRACT

The scope for portable microprocessor based measurement systems is, indeed, vast. However, the constraints on such systems are fairly severe in terms of both hardware and user friendliness. A previously expounded concept, the Roving Slave Processor, is developed to the point where comparatively demanding problems may be tackled. In particular, the problem of detecting the presence of a known signal in time variant noise is considered in depth and a novel solution proposed which combines adaptivity during the course of an experiment with ease of problem definition and subsequent solution.

CONTENTS

PART TWO

1. INTRODUCTION

The City University's Computer Aided Measurement (CAM) research group has been involved with many aspects of CAM from complex experiments employing a fast, powerful minicomputer through to less demanding problems within the capabilities of the early microprocessors. Amongst the more recent problems addressed by the group are:-

i) Making the skills and techniques of CAM available to those who need them, but do not have the necessary computer expertise.

ii) Harnessing the power of microprocessors for field based experiments as opposed to restricting CAM methods to the laboratory.

iii) Improving microprocessor performance by examining dual and multi-processor architectures

iv) Attempting to solve what must be one of the most difficult measurement problems within the scope of currently available microprocessors; the detection of the distinctive spike and wave pattern occurring in the EEG of epileptic subjects.

The four points above are covered, to a greater or lesser extent, by the work contained in this thesis.

The specific problem of non-volatile store for portable systems is covered in some depth with an adequate working solution provided. A recommendation for a more comprehensive solution to the problem is also given.

An advanced portable "black box" instrument is proposed based on a dual F100-L microprocessor system. However, the architectural details of the system and its performance are covered in only as much detail as required by the rest of the work contained in this thesis. The study of multiprocessor architectures is a major project in itself and has not been neglected by the CAM research group. Work on this aspect is detailed by Mainwaring-Samwell (34) and is complementary to the work described here. The two projects - "define a system" and "provide an application" - ran concurrently.

This creates an obvious problem in that work on the application had to proceed without the availability of the system on which it was designed to run. Thus it must be remembered that the application described in this thesis can only be judged on its potential performance and use. Possible variations which may be expected in the final system are discussed.

What, then, of the measurement problem itself? It has already been stated that the detection of the epileptic precursor in human EEG's is a very difficult problem even though it does not contain frequencies above 70Hz and therefore potentially within the range of microprocessor system applications. The problems lie within the EEG signal, which for this purpose is considered as noise. In particular an EEG is:-

i) Non-white

ii) Non-stationary (as it varies with the subject's state of rest)

iii) Frequently swamped by high amplitude noise due to muscular activity (artefact)

This specific problem is, though, just one of a class of measurement problems known as pattern recognition. Generally speaking all pattern recognition problems exhibit some or all of the following characteristics:-

i) The signal (pattern) to be detected is reasonably well known

ii) The noise is inadequately described

iii) The noise is non-stationary

iv) Specific solutions are time consuming to provide

Hence the strategy adopted was to develop a generalised signal detector and then apply it to the specific problem of the detection of the epileptic precursor.

It is recognised that the work described here has two distinct parts and, therefore, the thesis itself has been divided to provide distinction between the systems and the application work. Part One details the system components

and architecture with Part Two defining the measurement problem and proposed solution. Part Three ties together the first two parts and provides overall conclusions and suggestions for further work.

As a deliberate policy the references have been kept to a minimum. Standard texts have been chosen not only for their content, but also as indirect references to more specialised texts.

PART ONE

2. PORTABLE MICROPROCESSOR BASED MEASUREMENT SYSTEMS

2.1 MICROPROCESSOR BASED MEASUREMENT SYSTEMS

The advent of fast powerful microprocessors has broadened the horizons of those people working in the field of computer aided measurements (CAM). Until recently microprocessors were capable of performing only the simplest of measurement and control tasks, while the more demanding problems required the dedicated application of a minicomputer installation, such as that centred around the Ferranti FM1600B used by members of the CAM research group in the past (1, 2). With the arrival of more powerful microprocessors, the actual measurement to be performed, which is generally speaking highly demanding on CPU time, but not on peripherals, can be allocated to a small microprocessor system, leaving the minicomputer free to do what it is best at; that is file handling, extensive numerical analysis and assisting in the interpretation of results. The low cost of microprocessors and associated components allows a number of these pre-processors to be attached to one minicomputer system, thus increasing the number of real-time applications which may be performed concurrently.

In addition to the advantages gained by the low cost of microprocessors and their associated hardware there is something to be gained from the compactness offered by large scale integration (LSI). The small size and low weight of such systems allows the pre-processor to be portable in nature. There are a large number of measurements which require real-time processing and control, but are physically situated some distance from the immobile minicomputer. An insight into the problems of performing measurements over large distances may be obtained from Brignell (1). Quite simply the major problem is that the experimenter, who will be on site with the experiment, and the computer operator must communicate by telephone or similar device making human interaction in the experiment extremely difficult. There are also difficulties in transferring large amounts of data quickly over great distances whilst retaining integrity. The ability to take processing power on site overcomes these problems, particularly if the system is simple enough to use, so that the experimenter, who in general will not be a computing expert, can operate the device himself. These considerations, both economical and physical, led to the development of the roving slave processor (RSP) concept, which has formed the central part of the CAM Group's research since 1974 (3, 4, 5).

2.2 THE ROVING SLAVE PROCESSOR - A BRIEF OUTLINE

The concept of the RSP is well documented in the references quoted above and only a brief description will be given here. Basically, the RSP system consists of a host computer on which the off-line stages of the measurement are performed. This host computer is connected via a "breakable" link to one or more slave processors which will perform the actual measurement (see figure 2.1). The function of the slave is defined by standard software units compiled on the host computer, the program being "down-loaded" through the link. After the measurement has been performed the link may be re-connected to allow the "up-loading" of the experimental results for display or further analysis. The characteristics and requirements of the constituent parts of the RSP are discussed in the following sections.

2.2.1 The Host Computer

As mentioned previously, the host computer handles the off-line parts of the measurement which require the manipulation of both data and files. It must have sufficient processing power to perform the complex calculations often needed in the analysis of results (such as FFT's and correlations) and have the support of standard computer peripherals such as disk store, printer, plotter and terminals. Its operating system may be single-user or multi-user interactive, or even batch operated. It must also support high level languages as the program which compiles the standard software modules to define the instrument - the high-level definer (HLD) - is not only exceedingly complex, but should also be portable (in the sense that it may be run on various types of machine).

Some discussion within the CAM Research Group has meant that the development of the software has been somewhat slower than hoped. The original decision, which was to use the Ferranti FM1600B minicomputer, has been challenged on the basis that its operating system is particularly poor (although in other respects, e.g. processing power, it is quite suitable). In the present author's opinion the HLD itself should be written the most widely available high-level scientific language (FORTRAN) hence making the choice of host computer somewhat academic - one uses what is available. As the real-time software which forms the modules to be assembled by the HLD should also be portable (so that a variety of slave

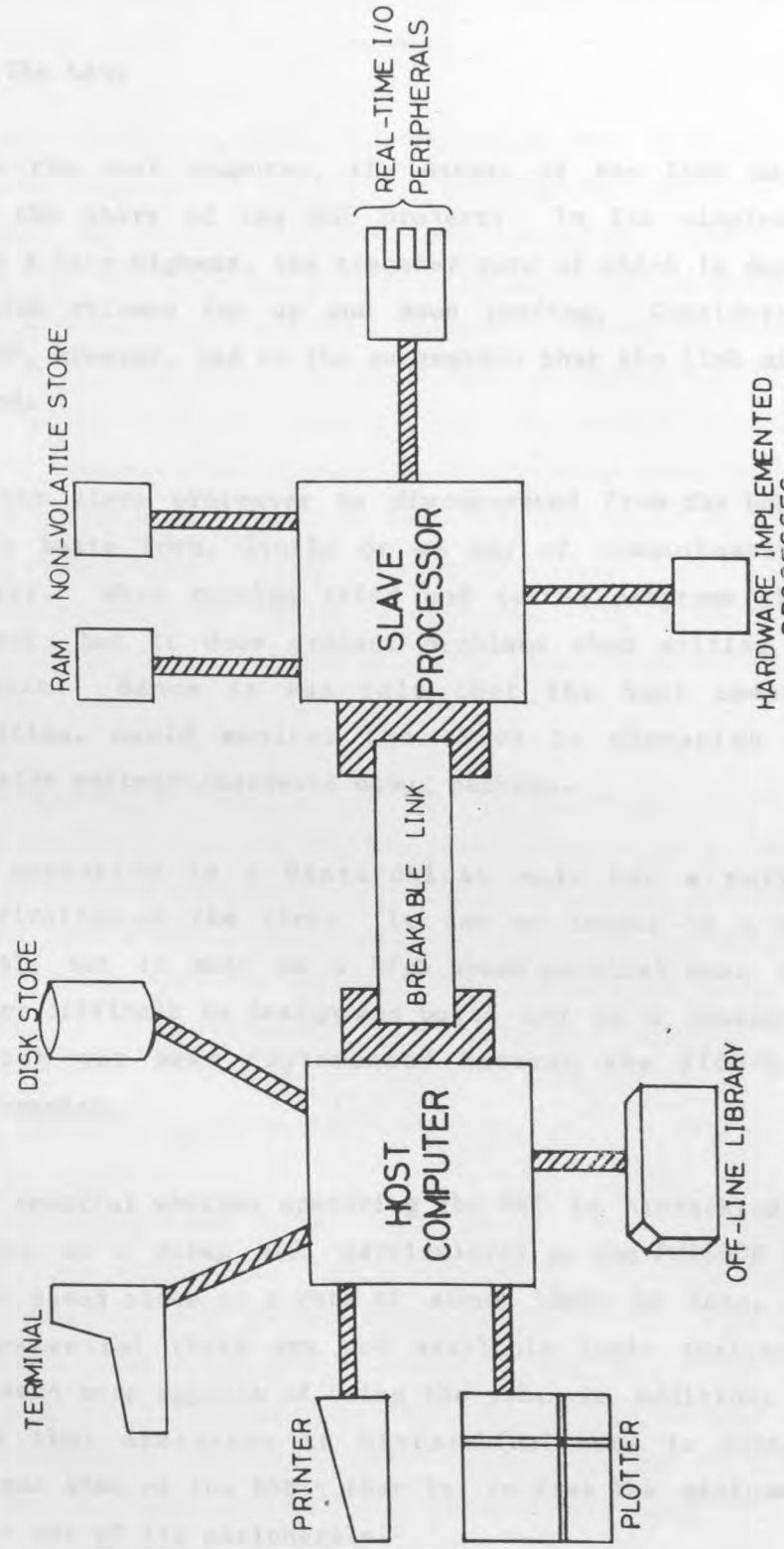


FIGURE 2.1 THE ROVING SLAVE PROCESSOR

processors may be used) they should be written in CORAL 66. Thus work could start immediately on the software even though it is not necessarily known what the host or slave processors are.

2.2.2 The Link

Unlike the host computer, the nature of the link has not been defined since the start of the RSP project. In its simplest form the link is merely a data highway, the transfer rate of which is dependent purely upon the time allowed for up and down loading. Considering the operation of the RSP, however, led to the suggestion that the link might have to be re-defined.

When the slave processor is disconnected from the host computer, it has, in its basic form, little or no way of communicating with its human operator. When running tried and tested programs this is not a serious drawback, but it does present problems when writing and debugging new software. Hence it was felt that the host computer, with all its facilities, could monitor the slave in operation thus providing an extensive software/hardware debug package.

This operation in a hierarchical mode has a serious effect on the specification of the link. It can no longer be a medium speed serial channel, but it must be a high speed parallel one. Such a channel would be very difficult to design and build and as a consequence this facility has not yet been implemented between the F100-L and the FM1600-B minicomputer.

It is doubtful whether operating the RSP in hierachical mode would be of any use as a debug aid, particularly as the FM1600B could only monitor a F100-L based slave at a rate of about 10MHz or less. If debugging aids are essential there are now available logic analyzers and logic state analyzers more capable of doing the job. In addition, the present author feels that operation in hierarchical mode is contrary to one of the original aims of the RSP - that is, to free the minicomputer CPU to allow better use of its peripherals.

If one excludes the possibility of operating the RSP in an hierachical mode, there does exist an alternative method of charging up the slave

processor with its program and retrieving data from it by a host computer by a method other than a "hard" data highway. This alternative method can be seen by examining further what happens to the data once it has been transferred to the slave processor.

After the slave processor's program has been compiled it is loaded from the host computer to the slave. In order for the data to be retained when the slave is disconnected, the data must be loaded into a non-volatile store. It has been decided, for reasons given in Chapter 3, that the non-volatile store will consist of a magnetic bubble memory unit. If the host computer itself were equipped with an interface to a magnetic bubble memory unit, then the program could be loaded directly into the non-volatile store, thus by-passing the loading stage to the slave processor. Similarly, data would be retrieved from the magnetic bubble memory unit by the host computer. It can be seen that this "soft" link does not actually need the presence of the slave processor, which may be of some use when the site of the experiment is a great distance away from the host computer.

A more immediate advantage is that the real-time software units, which are being developed on the F100-L development system, can already be loaded into the slave processor for final in-circuit testing, as the magnetic bubble memory unit interfaced to the slave processor also runs in the F100-L DOS development system. Such a system would also be reasonably fast, allowing the transfer of 11.5k bytes of data in just over 8 seconds.

2.2.3 The Slave Processor

The considerations affecting the choice of processor in a portable measurement system have been discussed by Comley (4) and Young (5). The requirement is for a processor with a 16-bit (minimum) word length with efficient interfacing and interrupt structures. These arguments hold for any portable microprocessor-based measurement system, as do the considerations given to other parts of the system, e.g. low power consumption, hardware/software trade-offs, etc. What is unique about the slave processor in the RSP is that, unlike most other systems, its program is loaded in a different place to where the measurement is performed. Also any data collected must be retained by the system until it can be up-

loaded to the host machine. It is, therefore, essential to have some form of non-volatile store. A block diagram of the slave processor system as it stands is shown in figure 2.2.

The design and the construction of the slave processor is almost completely dependent on the improvements made in technology. Hence although the idea of the RSP has been around for some time, it was not until 16-bit microprocessors were available that the first prototypes of the slave processors were constructed. These were based on the General Instruments CP1600 and the Ferranti F100-M, a MSI mock-up of a LSI microprocessor. Both prototypes were designed to be mobile rather than portable and were, therefore, fitted into a trolley mounted racking system. This allowed the inclusion of very heavy power supplies and other devices not part of the RSP concept, such as digital magnetic cassette drives and teletype interfaces. The measurement capabilities of the projected slave processors were tested on these prototypes and the feasibility of fairly advanced signal processing on portable microprocessors systems proved (see Section 2.3).

Neither of the prototypes were equipped with non-volatile storage for programs (the magnetic cassette on the CP1600 was used only to store data for convenience) although plans were made to use CMOS RAMS with battery back-up. Advances in other areas of technology (charge coupled devices and magnetic bubble memories in particular) have meant that the problem of non-volatile store can be solved, although CMOS RAMS still have a place in systems which are to be operated from a battery. The problem of non-volatile store is dealt with more thoroughly in Chapter 3.

The availability of a non-volatile store prompted the development of a second generation prototype which was truly portable. It is housed in an ordinary camera case which fulfils the specification of being roughly the size of a portable oscilloscope with a high degree of ruggedness. It was calculated that the total power consumption of the standard F100-L microprocessor and memory boards supplied by Ferranti plus a magnetic bubble memory unit with associated control and interface, an ADC/DAC and (if necessary) a teletype board was of an order such that a linear power supply of the right dimensions could be used. Indeed, it is quite possible that this system could just about be upgraded to include a dual F100-L system.

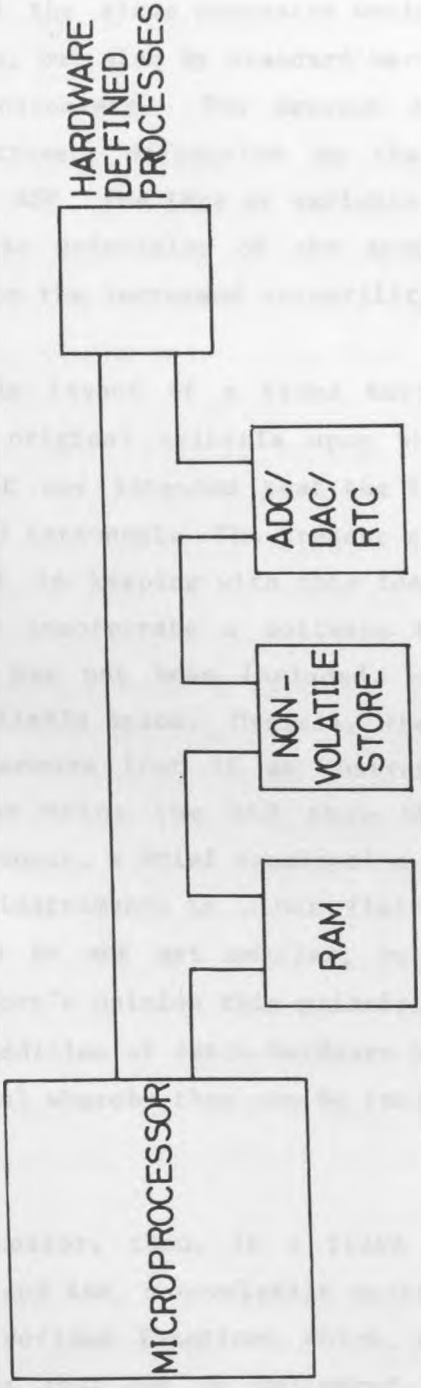


FIGURE 2.2 BLOCK DIAGRAM OF SLAVE PROCESSOR

Considering the actual architecture of the slave processor further it can be seen from figure 2.2 that there are only three mandatory component parts; the processor, its RAM and the non-volatile store. The other groups of components are the real-time I/O peripherals and special functions implemented in hardware. This led Comley (4) to suggest that the function of the slave processor would be defined not only by standard software modules, but also by standard hardware units which could be added or removed as necessary. The present author's view is that while the concept of instrument definition by the use of standard software is central to the RSP, the idea of variable hardware is totally at odds with some of the basic principles of the system, although there would be an obvious gain from the increased versatility.

The arguments in favour of a fixed hardware slave processor stem mainly from one of the original criteria upon which the RSP concept was based. Specifically, it was intended that the RSP could be used by unskilled (in computing terms) personnel. The present author feels that re-configurable hardware is not in keeping with this ideal, as it would be quite possible for the HLD to incorporate a software block that requires a piece of hardware which has not been included, or which cannot be included due to the lack of available space. Overall, the RSP must be of a high enough integrity to ensure that if an instrument is defined correctly by an unskilled person using the HLD then the slave processor should run properly. Moreover, a brief examination of the impact of higher levels of integration in instruments in other fields of measurement indicates that the instruments do not get smaller, but in fact become more complex. In the present author's opinion this principle should be applied to the RSP; that is, the addition of extra hardware units should occur when technology reaches the level whereby they can be included permanently into the slave processor.

The slave processor, then, is a fixed hardware device consisting of a microprocessor and RAM, non-volatile store, real-time I/O peripherals and some hardware defined functions which, when assembled, consumes only that amount of power that can be delivered by a small, lightweight power supply.

2.3 A GENERAL PORTABLE MICROPROCESSOR SYSTEM

In order to test the feasibility of a portable microprocessor based measurement system for the RSP, a prototype was developed which was of an acceptable size and weight. As it was not intended that this prototype should be used in the normal RSP configuration, it included several features not present in the slave processor as described above.

Initially, this prototype consisted of an F100-L microprocessor and 8k words of RAM. As it was not to be operated in RSP mode the non-volatile store was replaced by a method of entering programmes directly. As the system was housed in a case similar to the first prototype based on the F100-M (see section 2.2.3) this data entry system was a computer front panel, but this proved a time consuming and inefficient method and, therefore, a teletype interface was added. This meant a small monitor program (or operating system) had to be written and stored in U.V. erasable EPROMS. In addition the prototype also contained an ADC/DAC and real-time clock. A heavy duty power supply was installed in a separate box, so the whole thing was not particularly portable. It did, however, enable the first real signal processing programs to be tested and run. This system - with a software transient recorder program - was exhibited at the International Microcomputers, Minicomputers and Microprocessors (IMMM77) exhibition and conference in Geneva in 1977. With the inclusion of the heavy duty power supplies it was possible to upgrade the system to a dual processor configuration, for which the F100-L is particularly suitable (6). With one processor handling the I/O functions and the other performing the main processing it was possible to run a "continuous" pulse height analysis program not possible on a single processor system. This program was exhibited at the IMMM78 exhibition and conference. This system formed the basis for most of the work on dual processor systems, in particular the analysis of processing (and other) overheads incurred when employing close coupled processors.

There was still a need for a truly portable system and this was built into the camera case mentioned previously. By this time magnetic bubble memories were just beginning to become available and so a unit was added to the design. The magnetic bubble memory unit was not used to store programs for future loading into RAM (although it can if necessary) so the teletype interface was kept. Another addition was the introduction of a RAM based alpha-numeric CRT which, although not a part of the original RSP concept, proved a feasible method of

man-machine interaction in portable systems and, therefore, a distinct possibility in RSP applications. The prototype was powered by a small, linear design power supply contained within the camera case, thus making a truly portable system.

The camera case prototype was exhibited at IMMM79, running a transient recorder program similar to that exhibited at IMMM77, except for the additional I/O demands of informative messages to the CRT, storing previously captured transients on the magnetic bubble memory and recalling and displaying them on an oscilloscope. This truly portable prototype proved the extensive processing capabilities of portable microprocessor based instruments and that the RSP could be implemented using currently available technology.

2.4 CONCLUSIONS

The prototype developed provides the feasibility of powerful portable microprocessor based instruments, and thus add extra weight to the RSP concept. The possibility of dual processor slaves has also been proved, giving an added dimension to the RSP. One could envisage, then, two separate types of slave processor:-

1. A dual processor system, where high processing power is required as in I/O intensive applications.

2. A single processor system in which the added space vacated by the second processor and its associated hardware could be occupied by more hardware defined processes.

A third possibility is that the slave processor could be a low power system, with correspondingly less processing power, with the ability to run from a battery supply. This alternative has yet to be proved and, as with the very first prototypes, one must wait for the appropriate technology to become available. Nonetheless, such a system would be extremely useful and its feasibility should be established as soon as possible.

3. THE PROBLEM OF NON-VOLATILE STORAGE

3.1 INTRODUCTION

The need for a form of non-volatile store has been acknowledged since the beginning of the RSP project. The basic requirements for this non-volatile store is as follows:-

1. Small size and low weight for portability.

2. Low power consumption in order to reduce the size of the power supply and to increase the length of time that a battery operated system may function without recharging.

3. High reliability, as any malfunction within the RSP may go unnoticed until the slave is reconnected to the host computer.

4. Ruggedness to withhold the numerous knocks that a portable system is subjected to.

5. High speed so that the non-volatile store may form part of a medium to high performance measurement system.

6. High packing density to allow the collection of large amounts of data.

7. The ability to withstand changes in the electro-magnetic environment.

Points 1 and 2 are met by a variety of storage media, including electro-mechanical devices such as mini-cassettes. However, points 3, 4, 5 and 7 indicate the necessity for solid-state technology. Points 2 and 5 are, generally speaking, mutually exclusive, but as with most design problems a compromise is possible. Point 6 follows from point 1, and excludes the possibility of low power devices such as CMOS RAM.

By the first quarter of 1979 several new devices applicable to a non-volatile store unit had become available. In particular magnetic bubble memories (MBM) and charged coupled devices (CCD) were very promising due to their potentially

very high packing densities. As both MBM and CCD are serially accessed devices they could not be used as the system's read/write memory, but as the desired amount of non-volatile store would exceed the address space of the available microprocessor, it was no disadvantage to incorporate them in a backing store structure, in the same way that disks are used. At the same time the 64k x 1 bit dynamic RAM was announced and as this was of the same order of packing density as CCD and MBM, it was also worthy of consideration although higher packing densities were not expected for some time.

A report comparing CCD, MBM and RAM with respect to their applicability in portable systems as non-volatile backing store was presented to British Rail in April 1979. This report is given in section 3.2.

3.2 BACKING STORE FOR PORTABLE MICROPROCESSOR SYSTEMS

The following is a report presented to British Rail in April 1979, which examines the comparative advantages in using CCD, MBM and RAM as backing store in portable microprocessor systems.

BACKING STORE FOR PORTABLE MICROPROCESSOR SYSTEMS

INTRODUCTION

One limiting factor to the processing power and range of applications of portable microprocessor systems is the storage of large programs and reasonable amounts of data during the period when the system is removed from a stable power supply. In the past programs have been stored using CMOS random access memories with battery back-up facilities, but the low packing densities of these devices gives a low upper limit to the amount of non-volatile store within the system.

In applications requiring more data storage the only alternative to CMOS RAM was, until recently, electro-mechanical devices such as mini-cassette or mini-cartridge. Of these two the mini-cassette is the most feasible due to its lower power consumption. With data capacities in excess of 60k bytes mini-cassettes adequately fill the role of backing store, although the achievable data rates of only 300 bytes per second may be considered insufficient. The most serious disadvantage, however, is that being electro-mechanical devices mini-cassettes tend to be bulky and, more important, unreliable. In order to improve performance and reliability a solid-state technology must be used, and three likely candidates are examined in greater detail. These are dynamic random access memories (RAM), charge coupled devices (CCD) and magnetic bubble memories (MBM).

RANDOM ACCESS MEMORIES (RAM)

The introduction of 64kx1 bit dynamic RAM has led to a re-appraisal of the technology's role in storage systems. Designed with the mainframe in mind the 64kx1 bit RAM is also a feasible alternative to mini-cassette in portable applications.

The major disadvantage of this type of RAM is that it is not only volatile (therefore it requires battery back-up during system power-down in order to retain data), but it is also dynamic. This means that an access performing a refresh function must be made at regular intervals. This interval is 4ms for the 64kx1 bit RAM. In order to keep data valid, therefore, not only the RAM but also some refresh logic must be run off the battery back-up supply. This refresh logic will generally consume more power than the RAM itself. For

example on standby a 1M bit memory (16 64kxl bit RAM) will require about 1.2 watts, of which less than 0.4 watts are directly attributable to the RAM. Note that this power comes from a single +5 volt supply.

The disadvantage of volatility and the dynamic nature of the 64kxl bit RAM are partly offset by the advantage, namely speed and simplicity. Although the truly random access nature would have to be sacrificed for a page oriented struture, once a particular page is selected then any one of the 256 bytes in that page may be accessed as RAM with a cycle time of 250ns. In addition pages may be accessed randomly giving negligible delays when changing the page being accessed. Further, the technique of use of RAM in a system is well known and simple and the saving in development time and cost may well outweigh the higher per bit cost in comparison to CCD and MBM.

CHARGE COUPLED DEVICES (CCD)

CCDs, like the 64kxl bit RAM, are volatile and dynamic devices with (at present) a maximum capacity of 64k bits. The similarity ends there though, as CCD are serially accessible and not randomly accessible.

64kxl bit CCDs are (in the cases of the Fairchild F464 and the Texas TMS 3064) arranged as 16 randomly addressable end-around shift registers, each of 4096 bits with clock rates of between 1MHz and 5MHz. This means that the time taken to access the first bit of a page is, on average, between 0.41mS and 2.0mS depending on the clock rate and between 0.82mS and 4.0 mS extra is required to read or write the page.

A diagrammatic representation of the internal structure of a typical device is show in figures 1 and 2. The technique used in each register is called serial-parallel-serial (SPS) architecture, as data is loaded serially into the input register and when this is full it is loaded in parallel into the storage registers. The process is reversed for output, data being loaded in parallel from the storage registers to the output register where it is accessed in serial form (figure 1). In the F464 there are effectively 64 parallel registers each of 64 bits, giving the 4096 bits required for each shift register.

Each of the 4096 bit shift registers is arranged as shown in figure 2. The registers are independently addressable and unless a write cycle is requested data are transferred from the output of the registers to the input. The parts

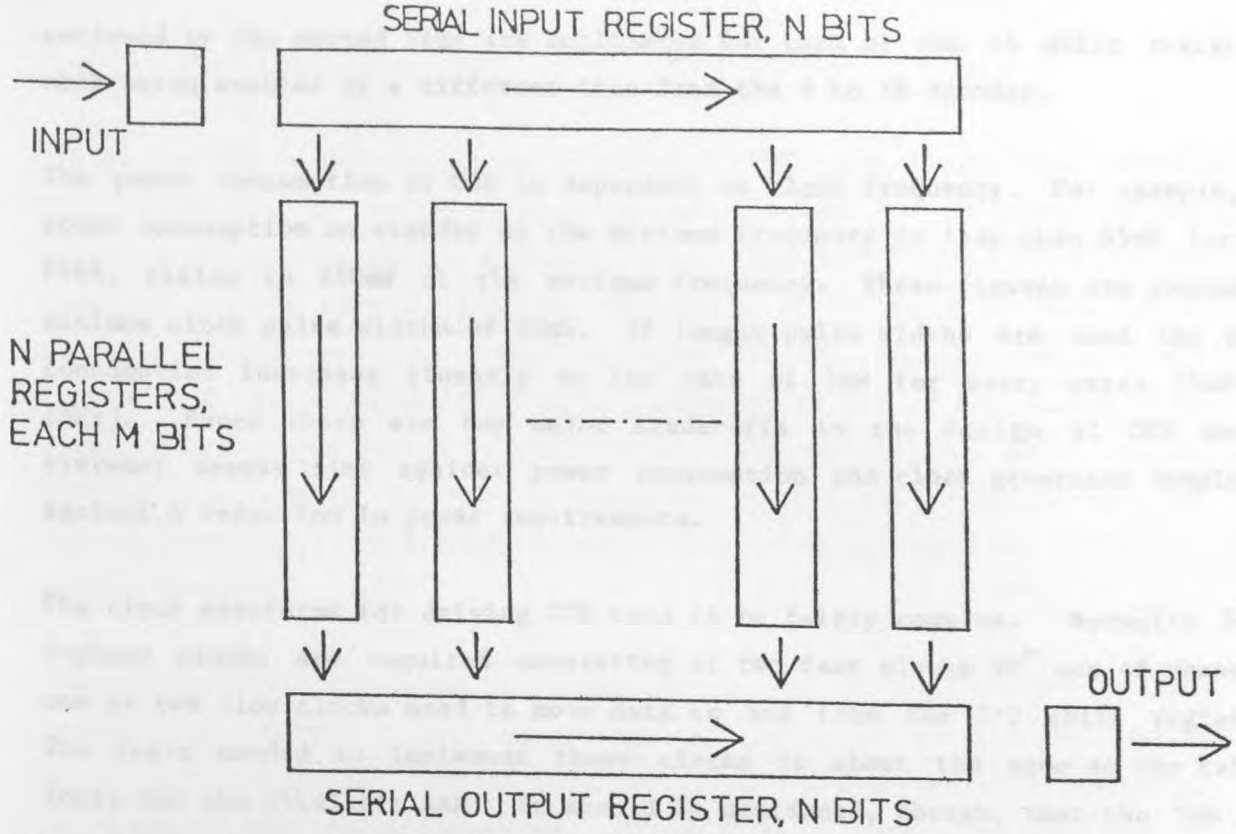


Figure 1. Diagrammatic Representation of a Shift Register Based on the Serial-Parallel-Serial Architecture.

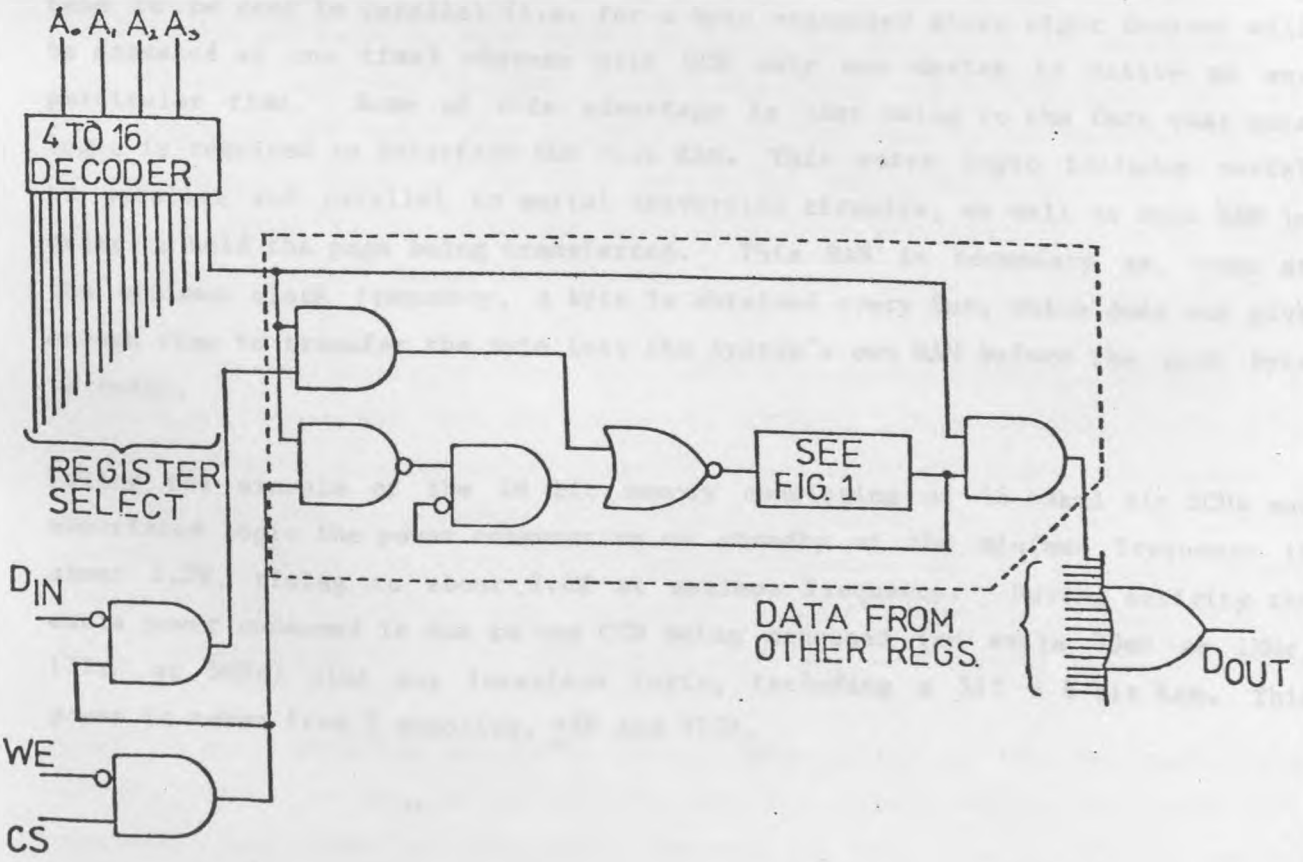


Figure 2. Internal Arrangement of F464 Charge Coupled Device.

enclosed by the dotted line are duplicated for each of the 16 shift registers, each being enabled by a different line from the 4 to 16 decoder.

The power consumption of CCD is dependent on clock frequency. For example, the power consumption on standby at the minimum frequency is less than 65mW for the F464, rising to 240mW at the maximum frequency. These figures are quoted for minimum clock pulse widths of 50nS. If longer pulse widths are used the power consumption increases linearly at the rate of 5mW for every extra 15nS (at 1MHz). Hence there are two major trade-offs in the design of CCD memory systems; access time against power consumption and clock generator complexity against a reduction in power requirements.

The clock waveforms for driving CCD tend to be fairly complex. Normally 3- or 4-phase clocks are required consisting of two fast clocks 90^{o} out of phase and one or two slow clocks used to move data to and from the I/O shift registers. The logic needed to implement these clocks is about the same as the refresh logic for the 64kxl bit RAM. It should be mentioned, though, that the TMS 3064 requires only a two-phase clock, which is a great deal simpler to generate.

When operational the CCD has a slight power advantage over dynamic RAM. This is because although both CCD and RAM have similar power consumption figures RAM tend to be used in parallel (i.e. for a byte organised store eight devices will be accessed at one time) whereas with CCD only one device is active at any particular time. Some of this advantage is lost owing to the fact that more logic is required to interface CCD than RAM. This extra logic includes serial to parallel and parallel to serial conversion circuits, as well as some RAM in order to hold the page being transferred. This RAM is necessary as, even at the minimum clock frequency, a byte is obtained every 8µs, which does not give enough time to transfer the byte into the system's own RAM before the next byte is ready.

Taking the example of the 1M bit memory consisting of 16 64kxl bit CCDs and associated logic the power consumption on standby at the minimum frequency is about 1.2W, rising to about 2.4W at maximum frequency. During activity the extra power consumed is due to one CCD being accessed (an extra 30mW at 1MHz, 132mW at 5MHz) plus any interface logic, including a 512 x 8 bit RAM. This power is taken from 3 supplies, $\pm$5V and +12V.

MAGNETIC BUBBLE MEMORIES (MBM)

Unlike the RAM or CDDs mentioned previously, MBMs are static and non-volatile. This means that no refresh circuitry is required and no power is needed in order to retain data during standby. MBMs also have higher packing densities than RAM or CCD with devices of capacities of up to 256k bits having been recently announced. The actual packages of MBM, though, are a great deal larger than the standard 0.3" d.i.l. packages of CCD or RAM, owing to the fact that a complete MBM device consists of not only the chip on which the magnetic bubbles are formed, but also a permanent magnet set to create the bias field and two orthogonal coils used to generate a rotating magnetic field to effect bubble movement.

Internally, MBMs are normally organised in one of two different ways. The simplest is a single loop or shift register (as in the devices made by Plessey) which is accessed as a single long serial in - serial out shift register. This arrangement has the disadvantage of very long access times, and so most manufacturers of MBMs have opted for an organisation known as major-minor loops (figure 3).

The major-minor loop structure is very similar to the serial-parallel-serial (SPS) technique used in some CCDs (see previous section). The difference is that the major loop does the work of both the input and output registers, and the length of the major loop is usually about the same as the parallel (minor) loops. Thus the length of the major loop is not necessarily equivalent to the number of minor loops. Also the major-minor loop structure is apparent to the user, unlike the SPS system used in CCDs.

It should be mentioned that there is a third alternative structure to single or major-minor loops and this is called the magnetic bubble lattice. Although so far only used in experimental devices (for example the MBM under development by IBM) and subject to problems of bubble propagation, the lattice has several advantages. With the lattice the bubble diameters can be reduced, giving higher packing densities, and the rotating magnetic field is not essential for bubble propagation, hence the orthogonal coils may be dispensed with.

Also different from RAM and CCD is the way MBMs appear to the designer. None of the inputs or outputs are TTL or MOS compatible, making the use of MBM a somewhat less appealing prospect. However, devices used previously in core

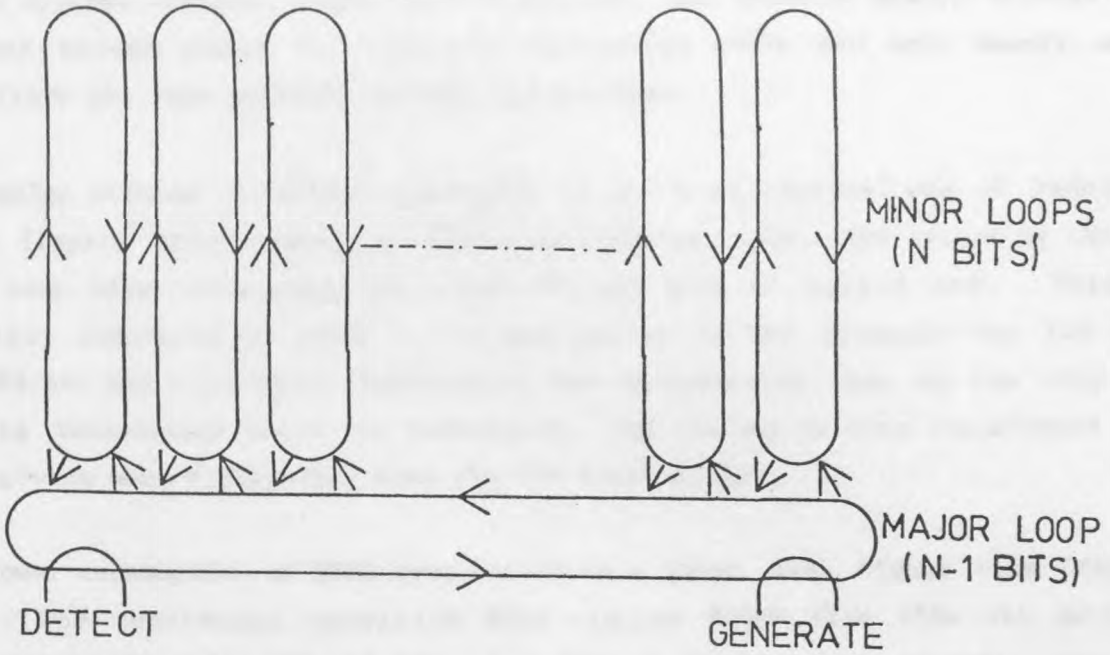


Figure 3. Basic Major-Minor Loop Structure.

memory systems are also usable in MBM systems, for example memory drivers can deliver enough power for the two orthogonal coils and core memory sense amplifiers are also suitable in MBM applications.

The design problem is further aggravated by the manufacturers' use of redundant loops (faulty minor loops) in order to increase yield. The effect of this is that some bits in a page are erroneous and must be masked out. This is normally achieved by using a bit map stored in ROM although the TIB 0303 254,688 bit MBM from Texas Instruments has a dedicated loop on the chip for storing information about the redundancy. The bit map is then transferred into RAM, giving more flexibility than the ROM based system.

The power consumption of MBMs when active is a great deal higher than CCDs or RAM. For continuous operation MBMs require 900mW (for 256k bit devices) compared to 400mW for CCD and RAM. For MBM of 64k bits or slightly more the active power consumption does come down slightly to between 450mW and 600mW. With the relevant support chips (coil drivers, sense amplifiers, controller, etc.) and other logic one can expect a MBM system to consume up to 11.5W when active.

MBMs are also slower than either RAM or CCD. The data rates range from between 45k bits/S up to 100k bits/S, though all devices are clocked at 100kHz. This gives average "first bit" access times of between 4ms (for devices of less than 100k bits capacity) and 7.3mS (for devices of about 256k bits). Note, however, that as MBM are static devices the page immediately following the one that is accessed is in the correct position to the accessed itself. Hence if contiguous pages are being accessed the "first bit" access time of all pages (except the first one) is the time taken for bits in the major loop to reach the detector system, which is about 0.9mS.

The page cycle time is very high and is the deciding factor in the speed of operation when pages are being accessed. This cycle time is approximately given by the clock period multiplied by the length of the major loop. This is usually between 5mS and 12 mS. This obviously affects the data rates adversely. For example, the TIB 0203 92k bit MBM, when accessing data, quotes a 50k bits/S data rate. Taking into account redundant bits this figure reduces to about 45k bits/S. However, if one considers a cycle time of 6.4mS, and that

144 bits are accessed during this time, the data rate drops to half of this (22.5k bits/S). This is obviously a great deal worse than the data rates of RAM or CCD, but it is better than mini-cassettes can achieve.

CONCLUSION

The inherent non-volatility of MBMs make them the ideal choice for backing store applications and what needs to be shown is whether RAM or CCD have enough other advantages over MBM to make the question of volatility insignificant.

In terms of per bit price CCDs are about as expensive as MBMs at the moment, but the higher packing densities of MBMs should make them cheaper in the future. Both MBMs and CCDs are less expensive than RAM. The problem of power consumption is a more complex question. Figure 4 helps the understanding of the varying relationships between the different technologies.

It is obvious from figure 4 that as MBM are accessed less frequently the power consumption tends to zero, due to the non-volatile nature of the devices. RAM and CCDs, however, tend to a power consumption value of what is required to retain data and keep the refresh circuitry operational. With a duty cycle of less than 10% MBM have better power consumption figures than RAM or CCD. It is expected that most backing store applications will yield duty cycle figures of less than 1% and so it would appear that MBMs have a definite advantage here. Figure 4, though giving an idea of what is happening as the store is accessed less frequently, does a great injustice to RAM and CCD. This is because the critical value of 10% corresponds to an average transfer rate of 2.2k bits/S for MBM, 100k bits/S for CCD at 1MHz, 500k bits/S for CCD at 5MHz and 30M bits/S for RAM. In order to obtain a true figure for power consumption the average data rate must be considered. This is done in figure 5.

Two points are obvious from an examination of figure 5. One is fairly surprising, that is CCDs at 1MHz have little or no advantage over RAM, and RAM is always superior to CCD at 5MHz. The second point to make is that below an average transfer rate of 2k bits/S, magnetic bubbles use less power than either CCDs or RAM. This figure is respectably high for the majority of portable applications. Consider, for example, the 1M bit memory. To read and write to the entire backing store requires the transfer of 2M bits. In order for the average transfer rate to be less than 2k bits/S, the system has to be passive for at least 1000S. There cannot be many portable microprocessor applications

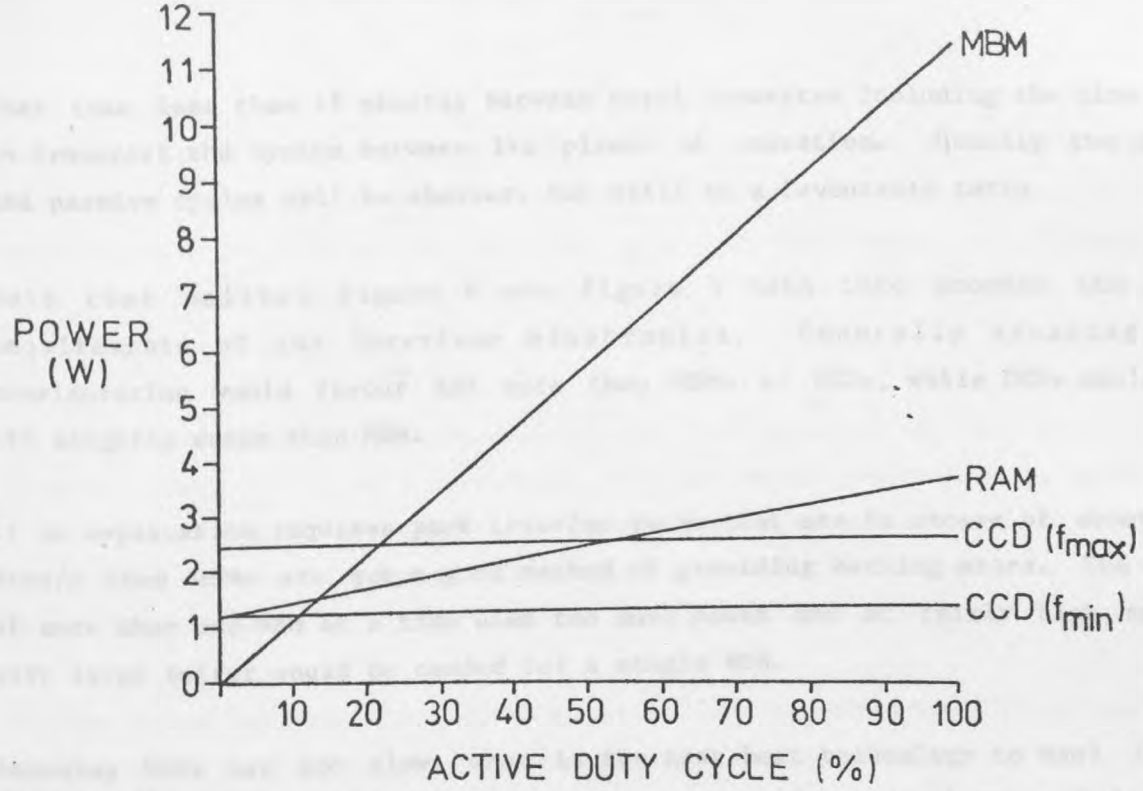


Figure 4. Average Power Consumption of 1M Bit Memory System Against % of Time When the Device is Active.

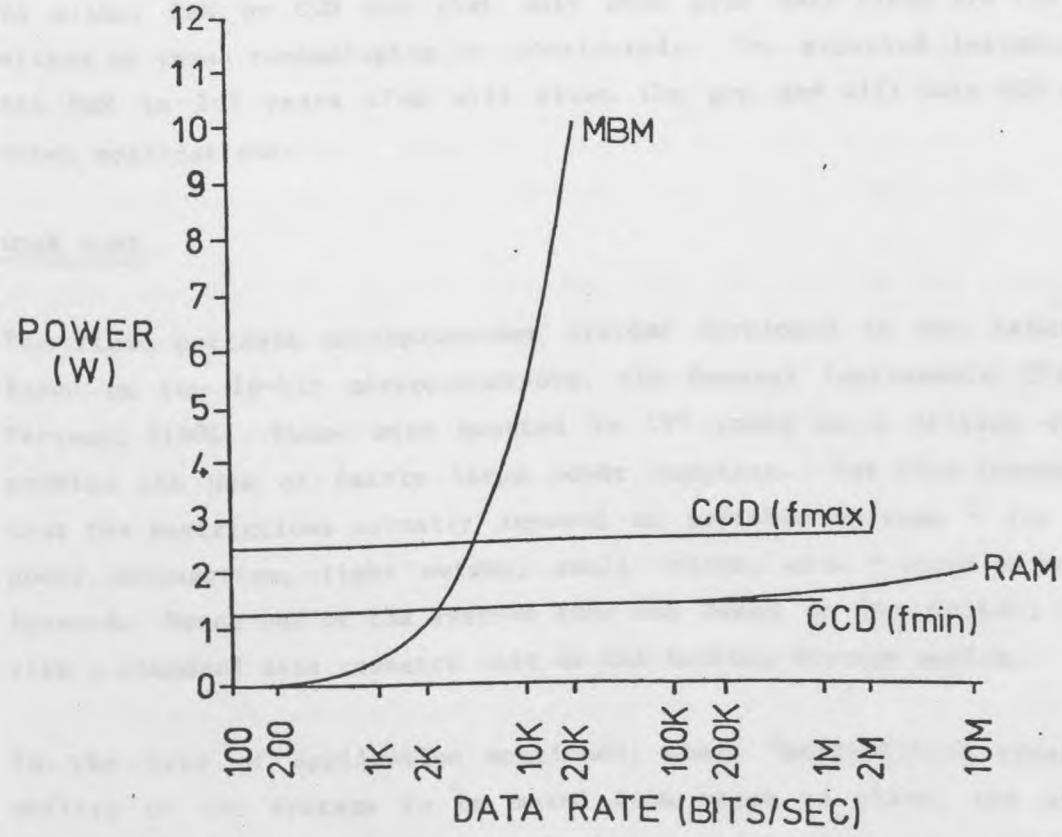


Figure 5. Average Power Consumption of 1M Bit Memory System Against Average Data Rate.

that take less than 17 minutes between total re-writes including the time taken to transport the system between its places of operation. Usually the active and passive cycles will be shorter, but still in a favourable ratio.

Note that neither figure 4 nor figure 5 take into account the power requirements of the interface electronics. Generally speaking this consideration would favour RAM more than MBMs or CCDs, while CCDs would come off slightly worse than MBM.

If an application requires peak transfer rates that are in excess of about 100k bits/S then MBMs are not a good method of providing backing store. The access of more than one MBM at a time uses too much power and at fairly high rates a very large buffer would be needed for a single MBM.

Assuming MBMs are too slow, what is the next best technology to use? In most cases the CCD clocked at its minimum frequency would be the better choice, but only on the basis of CCDs being cheaper than RAM. In any case, the trade-off betwen RAM and CCD is the standard speed vs cost.

Analysis shows, therefore, that MBM are superior in backing store applications to either RAM or CCD and that only when peak data rates are too high should either of these technologies be considered. The expected introduction of 1M bit MBM in 2-3 years time will widen the gap and will make MBM suitable for other applications.

WORK DONE

The first portable microprocessor systems developed in our laboratory were based on two 16-bit microprocessors, the General Instruments CP1600 and the Ferranti F100L. These were mounted in 19" racks on a trolley system, which enabled the use of fairly large power supplies. For this reason it was felt that the restrictions normally imposed on portable systems - for example low power consumption, light weight, small volume, etc. - could be to some extent ignored. Hence one of the systems (the one based on the CP1600) was equipped with a standard data cassette unit as the backing storage medium.

In the type of application mentioned, where "portability" simple meant the ability of the systems to be moved from place to place, the standard data cassette unit was an adequate solution to the backing store problem. One side

of a cassette could store up to 90000 characters at a rate of 630 characters per second (each character being 9 bits). It was apparent, though, that the size, weight and power consumption (over 15W) of the unit made it a totally impracticable proposition for the next stage of development, a microprocessor unit that could be carried by the user. The obvious alternative was to move to mini data cassettes.

The mini data cassette unit has none of the disadvantages of the standard data cassette unit. It is of a reasonable size and weight and the power consumption is less than 2W when operational. These advantages are gained at a loss of data capacity and a reduced data rate. Each side of the cassette may hold up to 60000 8-bit characters, with a transfer rate of 300 characters per second.

It soon became apparent that the slow data rate of mini data cassettes was a severe restriction on the range of applications. Added to this was the suspicion that the ruggedness and reliability of an electro-mechanical device may not be good enough for operation in hostile environments. In particular, it was felt that the magnetic tape itself, being only 1/8" wide, was likely to cause problems due to tape stretch, etc. In view of these objections - speed and reliability - the only solution was a solid-state technology.

Two major solid-state technologies were seriously considered, these being charge coupled devices (CCDs) and magnetic bubble memories (MBMs). A third alternative available at that time (the 16k RAM) was discounted on the grounds that a reasonable size backing store would require too much space.

Of CCD and MBM, the latter held the most promise for the future due to the higher possible packing density and its inherent non-volatility. However, CCDs were at a more advanced stage of development and were more readily available. Hence work was started on a Fairchild F464 64kx1 CCD and a Plessey 16kx1 MBM, with an order placed for two 64kx1 MBMs from Plessey when available.

Problems arose immediately in the use of the MBM. In particular, supplying the drive current to the coils posed serious problems. The two coils in the MBM have different characteristics and it was felt that the best method of driving the coils was to use external inductances in series with the coils in order to use a single +12V supply. In addition it was discovered that the devices used

to supply the drive current - core memory drivers - were not able to fulfil their intended function. This fact was borne out by Texas Instruments, who use an upgraded core memory driver on their own MBM system.

It became clear that the 64kxl MBM from Plessey would not be available for some time and so work started on the Fairchild CCD. It was felt that interfacing the CCD would be simpler than interfacing the MBM, as the CCDs inputs and outputs are either TTL or MOS compatible. It was found, however, that the four clocks necessary to operate the CCD were difficult to generate. Several designs were tried before a satisfactory CCD clock generator was made and this only works due to the compromises made, for example the CCD does not work at its maximum frequency nor does it have minimum power consumption at its working frequency.

In the last quarter of 1978 two developments altered our ideas about the backing store problems. The first was the announcement of 1/4M bit MBMs and this prompted a move towards the Texas Instruments devices.

The second development was the arrival of the 64kxl dynamic RAM and this makes it a serious contender to the CCD. It is not considered necessary to study the 64kxl RAM in any great detail and as enthusiasm for CCDs, apart from certain specialised applications, is on the wane, it is anticipated that the next stage of research will concentrate solely on the MBM.

PROPOSALS FOR FUTURE WORK

Two working MBM systems, both based on the Texas Instruments 92k MBM, have been constructed in our laboratory. One is interfaced to an F8 8-bit microprocessor and the other to an F100 16-bit microprocessor. Both are prototypes of a portable microprocessor system. It is proposed that extensive test runs be made on the MBM in an attempt to analyse the types of errors obtained and the frequency at which they occur. The results from the test runs will enable a decision to be made as to what error detection techniques should be used, the extent of error detection and whether error correction should be included.

A MBM system with appropriate error correction/detection would then form part of a microprocessor system designed with consideration given to fault tolerance and fault diagnosis. Such a system would also be influenced by the results of a study into diagnosing faults in other system components, including the

microprocessor, without excessive hardware overheads. Other workers have already examined these problems in a system's main memory and it is not envisaged that a great deal of work would be necessary in this area.

It is hoped that the final system would be run under operational conditions and that the effects of fault tolerance and diagnosis on system availability could be determined.

3.3 FURTHER COMMENTS

Since the report presented in Section 3.2 was written the complexion of the memory market has changed dramatically. The CCD market proved to be in specialised applications which require large shift registers of reasonable speed and the arrival of a 256k bit device was met with little enthusiasm.

Conversely, the demand for 64k bit dynamic RAMs is, as one would expect, enormous. The sheer volume of production of these devices will ensure that per bit costs remain extremely competitive. However, the technology itself would seem to be approaching its maximum in terms of packing density. This is shown by the delay between the announcement and the commercial availability of the 64k bit device and the absence (at the time of writing) of any immediate leader at the 256k bit capacity level.

MBMs have had a more chequered career. When the report presented in Section 3.2 was written the interest in MBMs was extremely high and for good reason (references 7 to 11). The report clearly indicates a number of advantages of MBM technology over competitive devices. However, within three years interest in MBM technology had diminished so drastically that, with the exception of the market leader at that time (INTEL), all the major manufacturers had withdrawn support for their devices. The lack of interest stemmed from the inability of MBM technology to compete against the removable flexible disk as the MBM manufacturers suggested. However, two major markets emerged which led to renewed interest in MBMs by early 1983:-

i) the increasing use of fixed disks, based on Winchester technology

ii) the office automation requirement of a large capacity central storage unit of a reliability not possible with electro-mechanical devices

This renewed interest has led to the imminent arrival of a 4M bit MBM from INTEL. This is an eminently suitable building block for a backing storage unit in a portable microprocessor system such as the RSP.

3.4 CONCLUSIONS

As mentioned previously a 92k bit MBM has been interfaced to the F100-L RSP prototype and has been subjected to extensive reliability tests (see Chapter 10). As an initial step 92k bits of non-volatile backing store is adequate, but only in the short term. As the complexity of application on the RSP increases greater demands will be made on the backing storage unit and an upgrade will be necessary. At least three alternatives exist:-

i) a simple circuit board rework to enable 4 92k bit MBMs to be used (the maximum for 1 controller)

ii) an upgrade to the currently available 1M bit INTEL MBM

iii) an upgrade to the 4M bit INTEL MBM when available

In the present author's opinion option iii) should be the ultimate target with either option i) or ii) being adopted in the short term to overcome any capacity problems occurring prior to the arrival of the 4M bit MBM.

4. AN ADVANCED PORTABLE MEASUREMENT SYSTEM

4.1 INTRODUCTION

Many of the problems of real-time computer aided measurment arise from the necessity to maintain data rates. In real-time computing the roles of the processor and its peripherals are reversed, as the processor must respond to the demands made by the real-time peripherals else data may be lost. In applications with a large amount of I/O this means the processor is constantly diverted from its main task in order to handle the I/O functions.

Obviously, the processing power of the processor is inextricably involved with the data rates. As data rates increase less time is available for processing. At the other extreme, a specific process will dictate the maximum permissable data rate. As an example, consider the process of implementing a recursive (or infinite impulse response, IIR) digital filter.

It is possible to implement any IIR digital filter by the means of cascading biquadratic elements, for example:

$$Y(z) = k\frac{\Pi\ (Az + Bz^{-1} + Cz^{-2})}{\Pi\ (z + Dz^{-1} + Ez^{-2})}\ X(z) \qquad 4.1$$

Each biquadratic element becomes, in the time domain,

$$y\ (T) = Ax(T) + Bx(T-1) + Cx(T-2) - Dy(T-1) - Ey(T-2) \qquad 4.2$$

In most filters (low-pass and high-pass, for example) A and C will be unity and B will be -2, which may be implemented digitally by a shift instruction. This means that two multiplications are needed to implement each biquadratic element. A further final multiplication is needed to adjust the gain. In the microprocessor used in the RSP, the F100-L, there is no multiply instruction, hence these multiplications must be implemented by software. With multiplication times of about 700mS a ten-pole digital filter would take about 7.8ms to complete the calculation required for each input sample. This gives a maximum sampling frequency of less than 130Hz. If these multiplications were performed in hardware data rates of up to 2 kHz could be expected.

One further technique of maintaining data rates would be to provide a second processor to handle the I/O. Not only would this allow the main processor to concentrate on its primary task (albeit with a small communication overhead), but some preprocessing could also be performed on the incoming data.

At the time of writing two prototype systems were in existence. The first is a portable RSP, based on a single F100-L microprocessor. The second prototype arose from the appreciation of the I/O bottleneck and is a dual F100-L system. These two systems are discussed below, along with a proposed third prototype of an advanced RSP.

4.2 THE SINGLE PROCESSOR PROTOTYPE

Housed in a photographic equipment case measuring 39 x 20 x 31 cm, the so-called "camera-case" single processor prototype was the first truly portable version of the RSP build by the CAM research group.

A diagram of the system is shown in figure 4.1. It consists of a F100-L microprocessor, 8k words of RAM, an ADC/DAC, 92k bits of non-volatile store and a memory-mapped display unit. In addition a teletype interface was included because at the time of writing there was no link to a host computer and so a method of loading programs was necessary. A small monitor program needed for the teletype is included in a U.V. erasable PROM. This also allows the monitoring and altering of locations in RAM in the absence of a front panel.

An indication of the portability and processing power of the camera case prototype is given by one application of the system demonstrated at the International Microprocessor, Minicomputer and Microcomputer Exhibition and Conference in Geneva in 1979 (IMMM79). The prototype was used to run an audio-frequency transient recorder program, displaying the results by continuously writing samples to the DAC for input to an oscilloscope. Previously captured transients were stored on the MBM unit for subsequent display. At the same time informative messages were output to the display unit.

Although obviously a fairly powerful system, the camera case prototype does have some undesirable features. The power supply, which provides 5A at 5V and 1A and -12V, is barely adequate, but a larger power supply would be unacceptable. A solution would be to use switching mode power supplies which would provide to 20A at 5V and 3A at -12V in roughly the same volume.

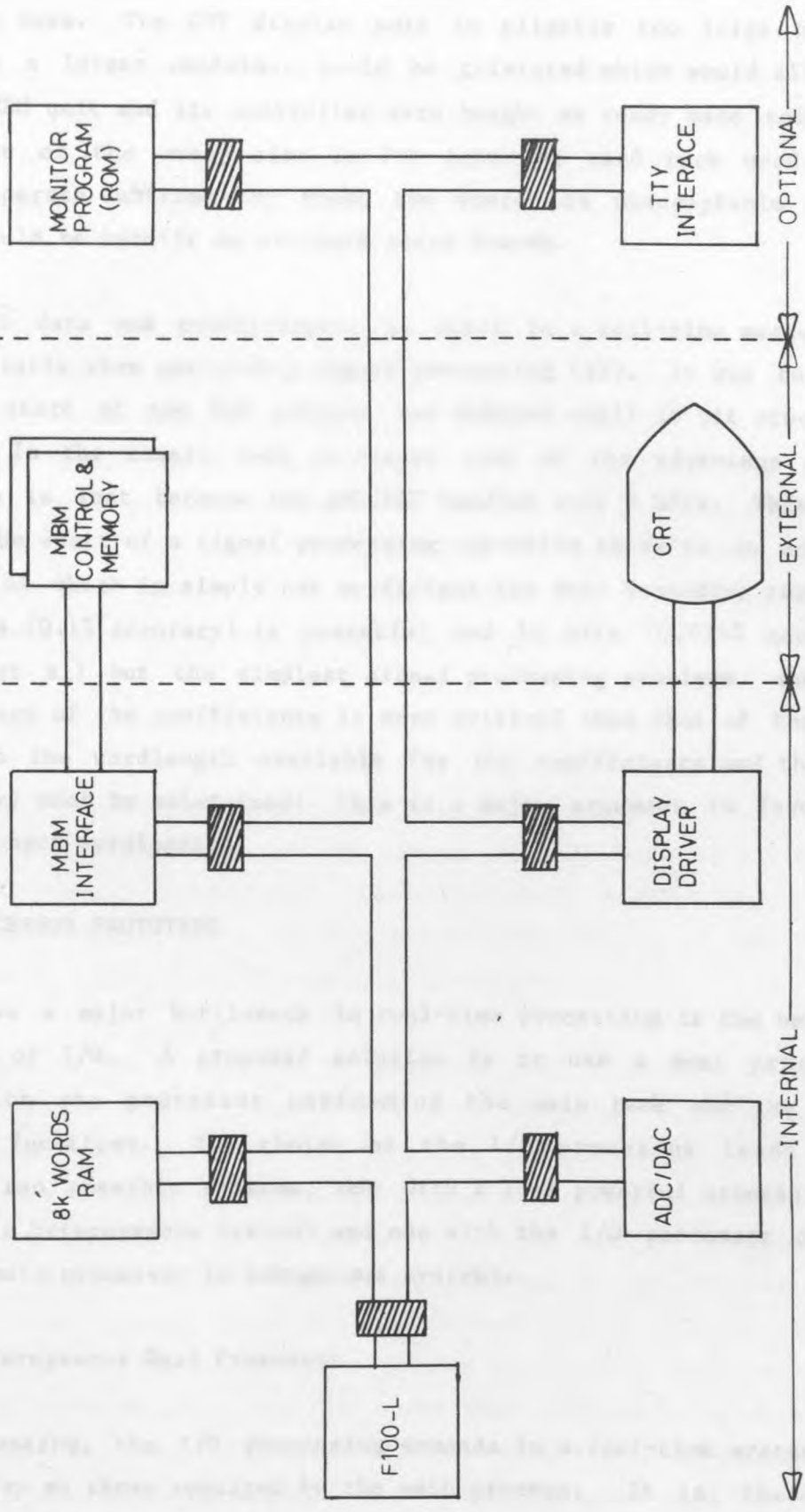


FIGURE 4.1 THE CAMERA CASE PROTOTYPE

Another unacceptable feature is that parts of the system are not contained within the camera case. The CRT display unit is slightly too large for the camera case, but a larger container could be tolerated which would allow its inclusion. The MBM unit and its controller were bought as ready made units and consequently are of the wrong size to fit into the card rack used. The inclusion of a separate subframe for these two boards is unacceptable and so the MBM units should be rebuilt on standard sized boards.

Accuracy, of both data and coefficients, is vital in a real-time measurement processor particularly when performing signal processing (12). It was for this reason that the start of the RSP project was delayed until 16 bit processors were available. In the camera case prototype some of the advantage of the longer wordlength is lost because the ADC/DAC handles only 8 bits. This means that right from the start of a signal processing operation there is an accuracy of only about 0.4% which is simply not sufficient for more demanding tasks. A minimum of 10 bits (0.1% accuracy) is essential and 12 bits (0.025% accuracy) is recommended for all but the simplest signal processing problems. However, because the accuracy of the coefficients is more critical than that of the data the ratio between the wordlength available for the coefficients and the data (2:1 approximately) must be maintained; this is a major argument in favour of processors with longer wordlengths.

4.3 THE DUAL PROCESSOR PROTOTYPE

As mentioned above a major bottleneck in real-time processing is the handling of large amounts of I/O. A proposed solution is to use a dual processor configuration with one processor performing the main task and the other handling the I/O functions. The choice of the I/O processor leads to a consideration of two possible systems, one with a less powerful processor for the I/O handling (a heterogenous system) and one with the I/O processor of the same type as the main processor (a homogenous system).

4.3.1 The Heterogenous Dual Processor

Generally speaking, the I/O processing demands in a real-time system will not be as heavy as those required by the main process. It is, therefore, possible to allocate the I/O processing to a comparatively simple processor. Any action taken by the I/O processor would be triggered by either the peripherals or the main processor and at all other times it

would remain dormant. It would not be practicable to exploit any spare processing power of the I/O processor to perform pre-processing on incoming data, as this would vary from one measurement to another. The development of programs in two different languages in order to perform one task would impose a severe man-power overhead. Thus the I/O processor would run a fixed program and would effectively be a logic replacement device.

The advantages of such a system are that the main processor is relieved of activities not central to the task in hand while not incurring excessive communication overheads, simplicity of system design and low cost (compared to dedicated device controllers). The viability of the heterogenous dual processor system is shown by the microprocessor manufacturers, who include I/O processors in most of the powerful microprocessor families. It is also possible to buy microprocessors specifically designed to be used in I/O applications.

4.3.2 The Homogenous Dual Processor

Selecting the I/O processor to be of the same type as the main processor is an attractive proposition to the system designer for both a practical and a philosophical viewpoint. The electrical similarity of the processors buses allows ease of connection, particularly in the case of the F100-L which supports multi-processor configurations (6).

The commonality of the software of a homogenous dual processor means that the I/O processor could run a variety of programs allowing greater flexibility and the possibility of extensive pre-processing on incoming data. In order to gain maximum benefit from such a configuration the two processors would have to be close coupled, in this case by communicating via a shared area of memory. The techniques used in systems with shared resources in order to avoid disastrous conditions such as "deadly embrace" are well known, but the processing overheads incurred by these techniques are not. Thus the homogenous dual processor also provides a valuable test bed for further analysis of this situation, which is a prerequisite for expanding into the highly promising field of multi-processing.

4.3.3 The Prototype

The attractiveness of the homogenous dual processor as a solution to the real-time I/O bottleneck and the desire to investigate close-coupled processors in operation led to the development of a dual F100-L based prototype. However, the acceptance of a homogenous dual processor should not exclude the use of a heterogenous system, particularly in the camera case prototype mentioned above. The objective of producing an advanced measurement system, though, cannot be met by a heteroengous dual processor and, therefore, the main effort within the project has concentrated on the homogenous system.

The dual processor prototype is housed in the case originally used for the F100-M, which was a pre-production MSI version of the F100-L. Figure 4.2 shows a block diagram of the system. Each processor has 4k words of RAM, the main processor's memory being private with the I/O processor's memory common to both processors. The buses are joined by the use of 2 interface sets connected in bus extension mode (6). The I/O processor has access to the ADC/DAC. The front panel, teletype and the PROM containing a small operating system are on the main bus in order that they may access both blocks of memory.

The architecture of the F100-L demands that certain locations are available. In particular location 0 (the link stack pointer) and location 2048 (the processor start address) must be present. In additional locations 1 to 255, which are used in indirect addressing, are too valuable to lose. It is essential, therefore, that both processors see their own RAM as being located between 0 and 4097. This immediately presents a problem for the main processor, for if its own RAM is located between 0 and 4097 how does it access the I/O processor's RAM, which recognises precisely the same address range? The answer is to provide, within the bus extension unit, an address mapping scheme such that addresses between, say, 4096 and 8191 on the main bus appear on the I/O bus as 0 to 4095.

As with the camera case prototype to ADC/DAC converter handles only 8 bits of digital I/O, but in this case this is not a serious disadvantage as the idea of the dual processor prototype was to investigate the operation of the system rather than actually perform as a measurement instrument.

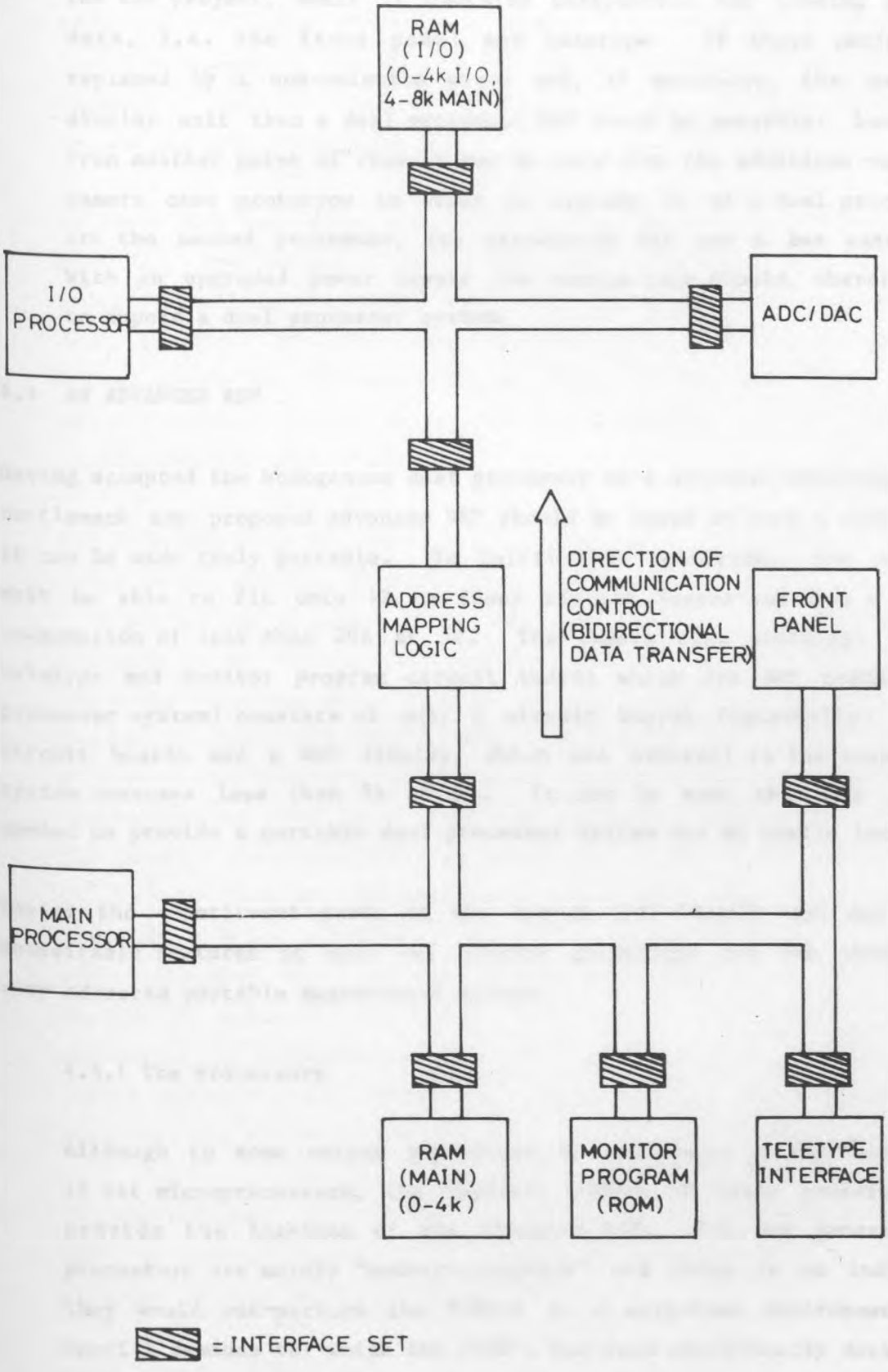


FIGURE 4.2 THE DUAL PROCESSOR PROTOTYPE

Initially, it was not envisaged that the dual processor would be used in the RSP project, hence it contains peripherals for loading programs and data, i.e. the front panel and teletype. If these peripherals were replaced by a non-volatile store and, if necessary, the memory-mapped display unit then a dual processor RSP would be possible. Looking at this from another point of view it can be seen that the additions needed on the camera case prototype in order to upgrade it to a dual processor system are the second processor, its associated RAM and a bus extension unit. With an upgraded power supply the camera case should, therefore, be able to suport a dual processor system.

4.4 AN ADVANCED RSP

Having accepted the homogenous dual processor as a suitable solution to the I/O bottleneck any proposed advanced RSP should be based on such a system, provided it can be made truly portable. To fulfil this requirement the entire system must be able to fit onto 12 or fewer circuit boards and have a peak current consumption of less than 20A at 5V. The camera case prototype (without the teletype and monitor program circuit boards which are not needed in a dual processor system) consists of only 5 circuit boards (internally) and 2 small circuit boards and a CRT display, which are external to the case. The total system consumes less than 5A at 5V. It can be seen that the little extra needed to provide a portable dual processor system can be easily included.

Taking the constituent parts of the system individually one may isolate the undesirable features of each and provide guidelines for the production of a very advanced portable measurement system.

4.4.1 The Processors

Although to some extent superceded by the advent of the "mini-on-a-chip" 16 bit microprocessors, the Ferranti F100-L is still powerful enough to provide the backbone of the advanced RSP. The new generation 16 bit processors are mainly "number-crunchers" and there is no indication that they would out-perform the F100-L in a real-time environment, with its special demands for which the F100-L has been specifically designed.

There is one major drawback to using the F100-L and that is the lack of hardware arithmetic. As shown in Section 4.1 this has serious effects on the processing power, and hence the range of applications, of a

measurement system. Fortunately the designers of the F100-L foresaw this problem and have provided a method of effectively increasing the instruction set of the F100-L by implementing external functions in hardware. This is achieved by operating the interface sets in special processing unit (SPU) mode (6).

Attempts have been made in the past to provide hardware arithmetic for the F100-L. The author's own undergraduate project included a hardware multiply unit for the F100-L and this was followed by a hardware divide unit designed by J.F. Cheesman (13, 14). However, neither of these attempts were totally successful and so it is fortunate that both designs were rendered obsolete by the introduction of the Ferranti F101 multiply/divide unit (MDU) (15).

The F101 MDU is a 40 pin d.i.l. device which implements a 16 x 16 bit fixed point multiply giving a 32 bit result and divides a 32 bit number by a 16 bit number to give a 16 bit quotient and a 16 bit remainder. It is designed to be electrically adjacent to the F100-L microprocessor before the first interface set. This has allowed a MDU and the two SSI devices needed to complete the interfacing to be implemented on a F100-L microprocessor board in "pick-a-back" fashion. When an external function instruction is read (i.e. when the function field F=0 and T=2 or 3) and the external function number is equivalent to one of the functions defined in the MDU then the MDU indicates to the F100-L that a multiply or divide function is being performed. The F100-L then idles until it receives an indication that the function is complete. The transfer of operands and results is achieved by DMA and any mixture of direct and indirect addressing is permitted. The instruction times vary from 11.0μS (single length multiply, all addresses direct) up to 18.7μS (double length division, all addresses indirect).

As the inclusion of the MDU is so simple and provides such an enormous gain in processing power it should be incorporated on both of the processor boards in the advanced RSP.

4.4.2 RAM Allocations

In the dual processor prototype both processors had 4k words of RAM, with the I/O processor's store also being available to the main processor. In addition some extra logic was required in order to map addresses 4096 to

8191 on the main bus to addresses 0 to 4095 on the I/O processor bus. The lack of private store leads to the possibility of program corruption by the main processor, while the added logic of the address mapping scheme is an unnecessary increase in hardware.

A simple solution to both problems mentioned above is to increase the amount of RAM on the I/O bus to 8k words thereby giving each processor 4k words of private store and a common area of 4k words. As the common area would appear to both processors as locations 4092 to 8191 the address mapping logic will become redundant.

It may be felt that 4k words of program for each processor is insufficient, and that RAM occupying the whole address space of both processors is desirable. This is possible to achieve using the standard Ferranti supplied memory boards as they allow the creation of "holes" within their address recognition range. The I/O processor RAM could then have a hole at the addresses recognised by the peripherals, and the main processor RAM would have a hole at the address range recognised by the bus extension unit. In any case increasing the amount of RAM in the system simply involves the replacement of one standard board with another of greater capacity.

4.4.3 Non-volatile Store

For operation in the RSP mode the advanced system must have non-volatile store in the form of a magnetic bubble memory unit. The camera case prototype includes such a unit, but in its present form it is not acceptable for the advanced RSP. In particular the MBM controller and memory boards are contained in a separate unit, and the capacity of 92k bits is inadequate for most measurement applications.

The MBM controller is capable of driving up to 4 of the 92k bit MBMs. This would give a total capacity of 36k bytes of non-volatile store, which is sufficient for a number of applications. Such a system could provide an adequate intermediate stage until the higher capacity MBMs were more readily available.

Although it may be possible, in terms of size, to include the MBM interface, controller and 4 memory units on one circuit board it would be advisable to make the non-volatile store a 2 board system. This would

have two advantages. The first is that, because the detection of a bubble gives only a 5mV reading, noise immunity for the memory units is imperative. The second advantage is that the memory unit could be interchangeable provided that the redundancy information for the defective loops were contained on the memory unit rather than on the interface/controller board. This would not be a particular advantage in a true RSP system, as vital information could be "up-loaded" to the host computer for storing on disk. It would mean, however, that the advanced RSP could, if necessary, operate in an autonomous mode.

4.4.4. Real-time I/O Devices

Both the camera case and the dual processor prototypes use the same type of ADC/DAC converter board. This provides two channels of which the digital side consists of 8 bits, the analogue input has a range of 0V to 2.5V and the analogue output has a range of 0V to 5V. Timing information is provided on the ADC/DAC with the processor extracting this information by polling a status word. Neither prototype has provision for digital I/O, but this would be quite simple to implement as memory-mapped ports.

The present ADC/DAC could be used as it stands in the advanced RSP and as other parts of the system (in particular the non-volatile store) require a fairly major re-design upgrading the ADC/DAC would be a low priority project.

There are good reasons, though, for improving the ADC/DAC for the advanced RSP. The accuracy provided by 8 bits is inadequate for a number of measurement problems, and even 10 bits would be barely sufficient. If possible 12 bit accuracy should be implemented on the advanced RSP, and more channels should be included in the design. The inclusion of digital I/O is also desirable, thus forming a general I/O system.

The method used for deriving timing information, i.e. by polling the status word, is particularly wasteful of processing time. Also, the sampling rate is set up in hardware either by altering the internal clock or by providing an external clock signal. A better solution would be to remove all the timing information from the ADC/DAC and provide another circuit board with a software programmable real-time clock (RTC) which

would generate interrupts. Apart from greater efficiency this method allows the setting of sample rates by software, a concept more in line with the RSP.

4.4.5. The CRT Display

The memory-mapped CRT display driver board is perfectly acceptable as it stands, and it would provide - along with a suitable manual input device driven from the same board - an efficient method of man-machine interaction for the advanced RSP. The only problem with the system is a mechanical one in that the CRT itself is too big to fit into the camera case prototype, but as the proposed advanced RSP would be installed in a different box it would be possible to define the dimensions of the housing to allow the CRT to be included. A case of similar dimensions to the camera case but with an extra 5 - 10cm height would be suitable. It may also be advisable to provide the CRT with a 12V supply independent of the main power supply to reduce the effects of noise generated by the RAM and the MBM

It should be noted that recent developments such as flat screen CRT's and (perhaps more important in portable systems) improvements in LCD technology may provide superior alternatives to the present standard CRT. However, this is outside the scope of this thesis.

4.4.6. The Overall System

A block diagram of a proposed advanced RSP is given in figure 4.3. In order to establish the portability of this system, it is instructive to compare it to the camera case prototype.

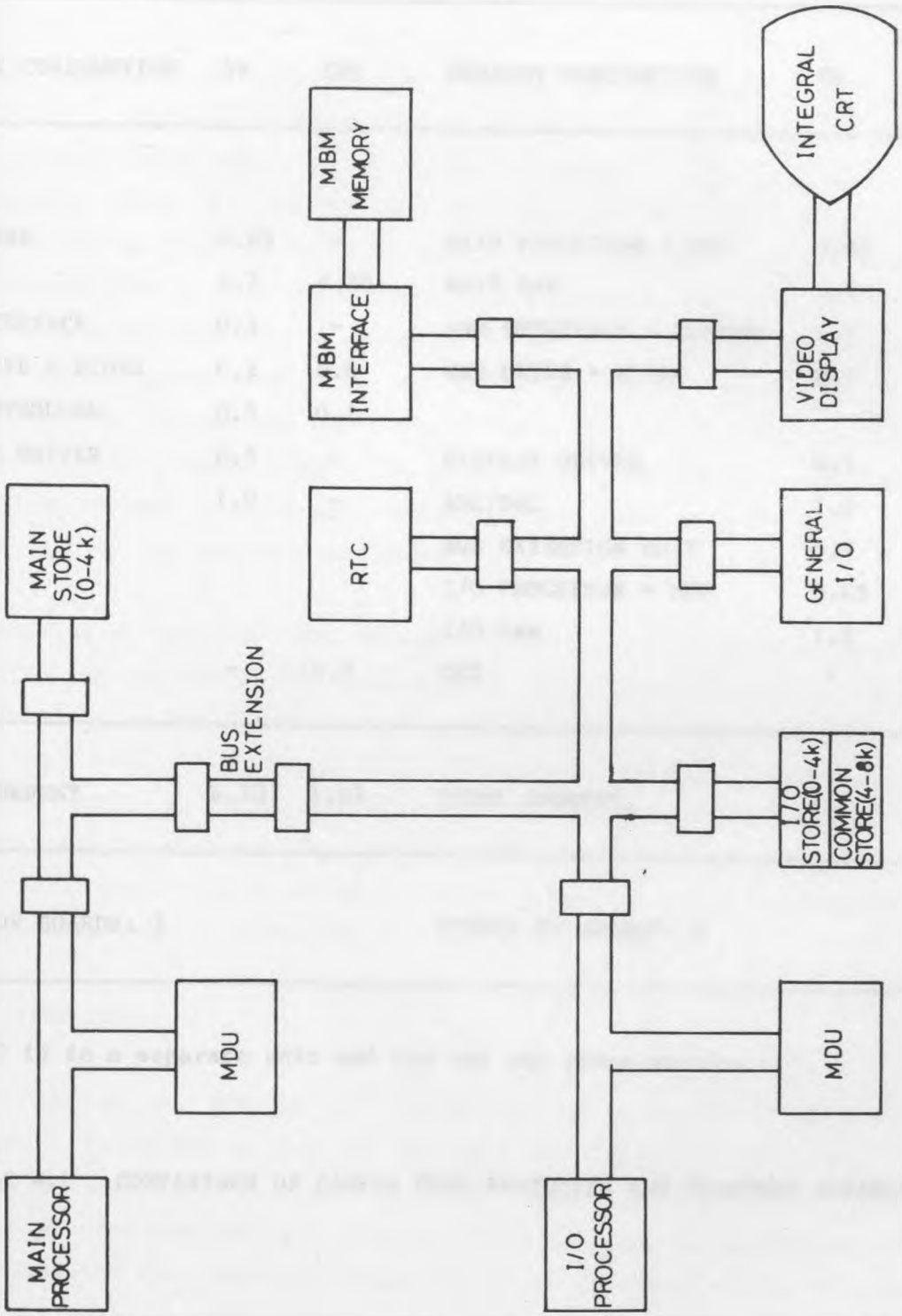


FIGURE 4.3 A PROPOSED ADVANCED RSP

CAMERA CASE			PROPOSED ADVANCED RSP		
CURRENT CONSUMPTION	5V	12V	CURRENT CONSUMPTION	5V	12V
PROCESSOR	0.83	-	MAIN PROCESSOR + MDU	0.85	-
RAM	1.2	0.25	MAIN RAM	1.2	0.25
MBM INTERFACE	0.5	-	MBM INTERFACE + CONTROL	1.0	0.1
MBM DRIVE + STORE	0.2	0.6	MBM DRIVE + STORE	0.2	0.6
MBM CONTROLLER	0.5	0.1			
DISPLAY DRIVER	0.5	-	DISPLAY DRIVER	0.5	-
ADC/DAC	1.0	-	ADC/DAC	1.0	-
			BUS EXTENSION UNIT	0.7	-
			I/O PROCESSOR + MDU	0.85	-
			I/O RAM	1.2	0.25
CRT*	-	0.9	CRT	-	0.9
TOTAL CURRENT	4.73	1.85	TOTAL CURRENT	7.5	2.1
NUMBER OF BOARDS: 7			NUMBER OF BOARDS: 9		

*The CRT is in a separate unit and has its own power supply.

TABLE 4.1 COMPARISON OF CAMERA CASE PROTOTYPE AND PROPOSED ADVANCED RSP

From table 4.1 it can be seen that the advanced RSP is comparable to the camera case prototype from the point of view of power consumption and size. In addition, both systems have identical current demands from a -12V supply of about 700mA.

The power in the camera case prototype is provided by a linear power supply. ("Linear" is a misnomer; a better name would be a series regulator power supply). It is now possible to buy switched mode power supplies which are smaller than this and that will deliver up to 20A at 5V and 1.75A at $\pm$ 12V in a single unit. This may seem insufficient for the advanced RSP in that the current requirement at +12V is 2.1A. However, it is considered desirable for the CRT to have a power supply independent of the system power supply as any noise on the +12V line (due to accessing either the RAM or the MBM) would cause an irritating flicker on the screen. If a small supply (1A at 12V) was provided by a separate unit the system current requirement would be only 1.2A at 12V, well within the capacity of the proposed switching power supply.

Selecting a case for the advanced RSP which is of approximately the same length and breadth as the camera case but between 5-10cm higher has two advantages. The first is that the CRT could be included within the case giving a self-contained system. The second advantage is that the circuit boards could be fitted in vertically, as opposed to horizontally as in the camera case prototype, thereby allowing up to 12 circuit boards within the system. Hence there is spare power and space to expand the advanced RSP if this is felt to be necessary.

4.5 APPLICATION

The basic aim of the RSP project is to provide a general, software defined instrument. Providing a list of specific applications, therefore, can only serve only as examples. However, the added power of the second processor in the proposed advanced RSP may not be necessary for all applications and so a brief analysis of the characteristics of those problems which do require the advanced RSP would be informative.

In simple terms the types of problem for which the proposed advanced RSP is most suited can be divided into two categories. The concept of the dual processor was derived from observing the bottlenecks in real-time processing and identifying I/O as a major problem. Thus an application requiring a large

amount of I/O handling will probably need the advanced RSP. Paradoxically, the second category of measurement problem for which the advanced RSP is ideally suited is where there is little I/O handling required, thus leaving a large amount of spare processing power in the I/O processor. If an application can be divided into two parts in an efficient manner then a greater range of tasks is possible by exploiting the I/O processor's spare power. The author's own work is such an application, and this is demonstrated in Chapter 8 where a signal detection system based on the advanced RSP is proposed. Another application is the implementation of digital filters, particularly when using cascaded biquadratics where each individual element may be calculated independently.

5. THE PROBLEM OF NON-STATIONARITY IN MEASUREMENTS

5.1 DEFINITIONS

A stochastic process (16) may be interpreted as a family of time functions $\underline{x}(t,\zeta)$ where ζ is a random variable (rv), and specifies a particular time function. In the case of the analysis of EEG's the stochastic process is the set of all possible EEG records, and the rv ζ would identify a particular subject from which an EEG record was taken. It is common, for the sake of clarity, to assume the dependence of the process on the rv and represent the process as $\underline{x}(t)$.

The mean (ensemble average) of a process is the expected value of the family of functions at a particular time:

$$\eta(t) = E\{\underline{x}(t)\} = \int_{-\infty}^{\infty} xf(x;t)dx \qquad 5.1$$

where $f(x;t)$ is the probability density function of the values of the family of functions at time t. The mean is, therefore, a function of time.

The autocorrelation is the joint moment of the process at two times, t_1 and t_2

$$R(t_1,t_2) = E\{\underline{x}(t_1)\underline{x}(t_2)\} = \int_{-\infty}^{\infty} x_1 x_2 f(x_1,x_2;t_1,t_2)dx_1\, dx_2 \qquad 5.2$$

A stochastic process is said to be stationary in a strict sense if the stochastic properties of the process do not vary with time. For this to be true the n^{th} order probability density function must obey the condition

$$f(x_1,x_2,..,x_n;t_1,t_2,..,t_n) = f(x_1,x_2,..,x_n;t_1+e,t_2+e,..,t_3+e) \qquad 5.3$$

for all values of e.

It is impossible to prove equation 5.3 for any non-trivial process, and so the concept of stationarity in the wide sense is more useful in practice. In this case a process is considered stationary if equation 5.3 holds for the 1st-order and 2nd-order probability density functions. Consequently, a process is considered stationary in the wide sense if

$$f(x;t) = f(x;t+e) \qquad 5.4$$

and

$$f(x_1,x_2;t_1,t_2) = f(x_1,x_2;t_1+e,t_2+e) \qquad 5.5$$

for all values of e.

Referring to equation 5.1, and bearing in mind equation 5.4, it can be seen that if a process is stationary in the wide sense then

$$\eta(t) = \text{constant} \qquad 5.6$$

Similarly, from equations 5.2 and 5.5 it can be seen that the autocorrelation can be expressed in terms of the time difference only, such that

$$R(t_1,t_2) = R(\tau) \quad , \ \tau = t_1-t_2 \qquad 5.7$$

Hence we can say a process is stationary in the wide sense if, and only if, the mean is a constant and the autocorrelation may be expressed as a function of the time difference. Hereafter, the word stationary will be taken to mean stationary in the wide sense.

A process is said to be stationary over an interval (T_1,T_2) if

$$\eta(t) = \text{constant} \quad , \ T_1<t<T_2$$

and

$$R(t_1,t_2) = R(\tau) \ , \ \tau = t_1-t_2 \quad , \quad \begin{matrix} T_1<t_1<T_2 \\ T_1<t_2<T_2 \end{matrix}$$

Finally, a process is said to be ergodic if the statistics of the process can be derived from a single member of the family of functions. This implies that the time averages of each member of the family of functions are equal to the ensemble averages given in equations 5.1 and 5.2. Obviously, a process which is not stationary cannot be ergodic. However, it does not follow that the

ergodic assumption can be made for a stationary process. Consider, for example, the stochastic process of the family of functions whose values are a constant such that

$$\underline{x}(t,\mathfrak{F}_i) = i \quad , \quad i \in \mathbb{R} \qquad 5.8$$

The process is obviously stationary, and the mean is 0. For a particular time function specified by $\mathfrak{F}_i$ the time average is i. Therefore, the time average and the ensemble average are not equal, and the process is not ergodic.

As ergodicity is difficult to prove it is often assumed, and this is usually valid if a process is stationary and the time average of each member of the family of functions is equal to the ensemble average. In addition, it is possible for a process to be ergodic with respect to some parameters but not others.

5.2 THE EFFECT OF NON-STATIONARITY ON INSTRUMENT DESIGN

In the real world all signals contain, to some extent, an unwanted noise component. For the purposes of discussion an instrument will be considered here as a system designed to extract the useful information from the unwanted noise.

It is imperative that the instrument designer has available to him some knowledge regarding the characteristics of both the information carrier (the signal) and the unwanted noise. Usually, the noise is considered additive, providing a composite working input x(t) such that

$$x(t) = s(t) + n(t) \qquad 5.9$$

where s(t) and n(t) are the signal and noise components respectively. The quality of an instrument is largely dependent on the amount of prior knowledge of the composite signal.

In order to examine the effect of non-stationarities it will be assumed that signal s(t) is completely defined and its characteristics known to the designer. This is not always the case in practice and this will complicate the design process. What is important here is how the quality of the noise effects the instrument design.

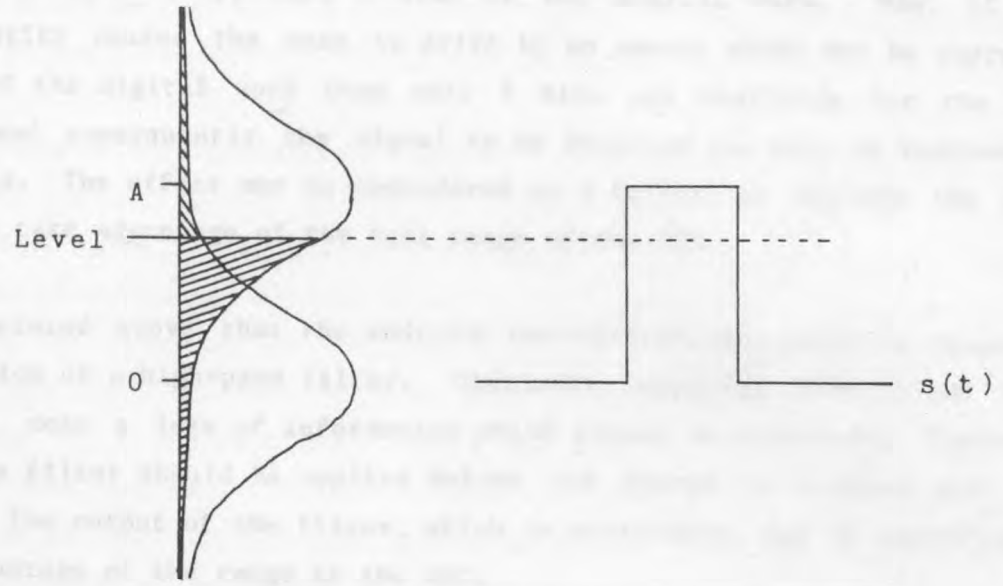
In its simplest form the noise will be deterministic, and therefore action may be taken to counteract its effect on the composite signal. For example, if it is known that the noise is, say, a 50Hz sine wave then all that is required is to remove the 50Hz component with a notch filter (bearing in mind, of course, the effect this may have on the signal). It is rare that a measurement problem is so simple, and it is usually better to consider the noise as a stochastic process, where only its probabilistic properties are known.

One important measurement problem, and one which forms the major part of the present author's work, is the detection of a known signal. As the signal is known, in some sense, it is necessary only to detect the presence of the signal rather than extract its shape. In such problems extensive knowledge of the noise component is essential. Furthermore, if the noise is described probabilistically then it is only possible to provide a probability of the presence of the signal, and therefore a good model of the noise component can give some information as to the probability of an error.

As an example, consider the detection of a pulse of known amplitude, A. If it is known that the noise is, say, zero mean Gaussian in nature then it is possible to determine the probability of an error when using a simple level detection scheme (see figure 5.1). Thus we can adjust the level in order to optimise the trade-off between the probabilities of false alarms and missed pulses. This system may be applied to signals in noise with other probability of errors determined in the same way.

Consider, now, the same measurement problem but with Gaussian noise with a mean value η. As can be seen from figure 5.2 the error probabilities are different from the case where the noise had a zero mean. In fact, to produce the same optimal trade-off used for the zero mean noise problem the level would have to be altered in the non-zero mean case by an amount η. If the mean value of the noise varies throughout the course of the measurement the composite signal forming the input to the instrument will be non-stationary. This form of additive non-stationarity is the simplest to deal with as it manifests itself as a low-frequency drift in the mean value and therefore it may be removed by a high-pass filter.

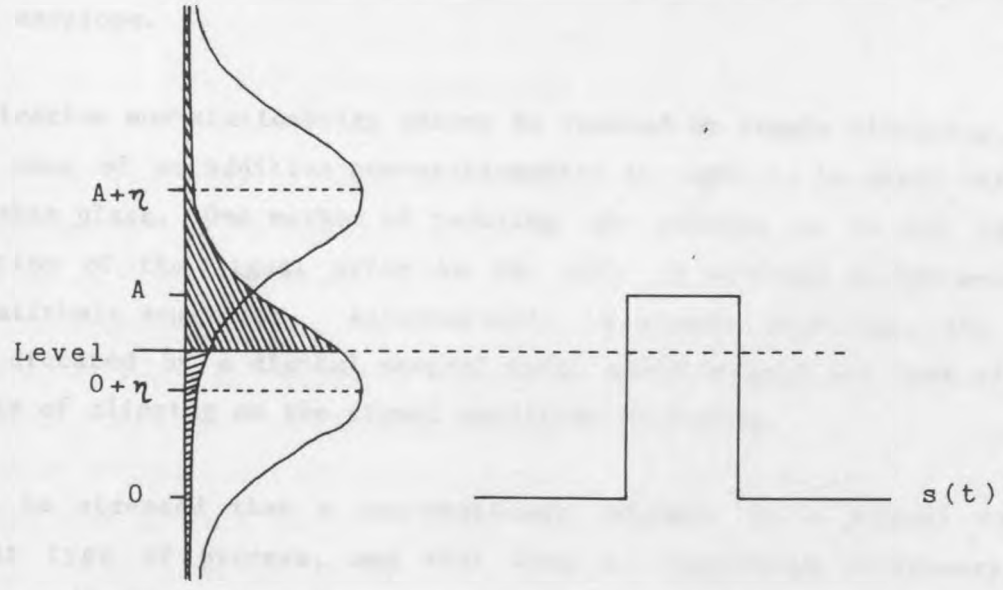
In a digital system the presence of an additive non-stationarity will cause a loss of accuracy in the ADC and consequently amplify the quantisation noise. To see why this is so, consider a composite signal as input to an 8-bit ADC. To avoid clipping, the 8 bits of the ADC must be sufficient to represent the



Area of ▨ = probability of s(t)=0 being read as s(t)=A

Area of ▨ = probability of s(t)=A being read as s(t)=0

FIGURE 5.1 PULSE INDICATION BY LEVEL DETECTION IN ZERO MEAN GAUSSIAN NOISE.



Area of ▨ = probability of s(t)=0 being read as s(t)=A

Area of ▨ = probability of s(t)=A being read as s(t)=0

FIGURE 5.2 PULSE INDICATION BY LEVEL DETECTION IN GAUSSIAN NOISE OF MEAN η.

whole range of the composite signal which may mean that the signal to be detected occupies only, say, 5 bits of the digital word. Now, if the non-stationarity causes the mean to drift by an amount which may be represented by 2 bits of the digital word then only 6 bits are available for the composite signal and consequently the signal to be detected can only be represented by 3 or 4 bits. The effect may be considered as a failure to amplify the signal in order to take advantage of the full range of the ADC.

It was stated above that the additive non-stationarity could be removed by the application of a high-pass filter. Obviously, applying this filter after the ADC will mean a loss of information which cannot be tolerated. Therefore, the high-pass filter should be applied before the signal is sampled and quantized so that the output of the filter, which is stationary, may be amplified to take full advantage of the range of the ADC.

A similar loss of information occurs in the case of a multiplicative non-stationarity. With this form of non-stationarity the range of values of the signal varies with time giving periods of "quiet" and "noisy" activity. As the ADC must cope with the noisy periods in the signal the range of the signal during the periods of quietness may occupy only a very few bits of the digital word. This may be considered as the signal being modulated by a relatively low frequency envelope.

A multiplicative non-stationarity cannot be removed by simple filtering, but as with the case of an additive non-stationarity it ought to be dealt with before the ADC takes place. One method of tackling the problem is to use non-linear amplification of the signal prior to the ADC. A suitable method would be to use a logarithmic amplifier. Alternatively, a linear amplifier, the gain of which is dictated by a digital control word, could be used but this allows the possibility of clipping as the signal amplitude increases.

It should be stressed that a non-stationary element in a signal is not a particular type of process, and that what is considered stationary in one problem may well be considered non-stationary in another. In the cases given above, where the non-stationarity is due to some low-frequency component in the noise, the type of process which would be considered non-stationary depends upon the nature of the measurement problem. For example, if a measurement is to be performed on a signal with frequencies that lie in the range 5Hz to 15Hz a 10Hz sine wave would not be considered a non-stationary element. However, if the range of interest was 5kHz to 15kHz then the 10Hz sine wave would appear as

a drift in the mean value of the signal, and therefore, it would be considered a non-stationarity. This subjectivity makes it very difficult to tackle the problem of non-stationarity in a general sense.

A third form of non-stationarity, which is somewhat more difficult to cope with than the types already mentioned, occurs when the quality of the process changes with time. This may manifest itself as a change in the emphasis of frequency components of the signal, or indeed any other parameter which describes the behaviour of the process. Obviously, a multiplicative non-stationarity is a sub-set of this form of non-stationarity, but as it may be dealt with in a comparatively simple way and has a detrimental effect on quantisation accuracy it was considered separately. For the purposes of discussion it will be assumed that the effects of any multiplicative or additive non-stationarities have been rectified prior to sampling. It is therefore assumed that the signal is represented to an accuracy dictated by the ADC with no loss of information due to non-stationarities.

Consider the problem of detecting a known signal in background noise as in Chapter 6. It can be shown (17) that the filter which optimises the signal to noise ratio, and therefore provides a best estimate as to the presence of the signal, must satisfy the condition

$$\int_o^T h_o(z)R_{nn}(\tau-z)dz=s(T-\tau),\ 0 \leqslant \tau \leqslant T \qquad 5.10$$

where $h_o(t)$ is the impulse response of the optimum filter, $R_{nn}(\tau)$ is the autocorrelation of the background noise and $s(t)$ is the signal to be detected. The filter which optimises the signal to noise ratio is known as the matched filter. Obviously, such a filter is only optimal for a given signal in background noise which has specific statistical properties. If the noise was non-stationary, so that its autocorrelation is dependent on time rather than just the time difference, it would not be possible to construct a matched filter which was optimal at all times. In general, an instrument which is designed to meet some performance criterion given noise with specific statistical properties will not necessarily meet that criterion should the properties of the noise change.

It is essential that the instrument designer has at his disposal a thorough knowledge of the properties of the noise component in the signal. This will not be the case for a non-stationary noise component unless the nature of the non-stationarity is well known, and for this reason there exist no design

methods directly applicable to non-stationary processes. The designer must, therefore, apply those techniques used to deal with stationary processes to problems containing non-stationarities. If the characteristics of a signal do not change appreciably over a period of time then it is possible to assume that the signal is stationary over that interval. Thus a (theoretically) infinite non-stationary process may be considered as a number of stationary processes of finite duration. Obviously, an instrument performing a measurement on the segmented input must be designed in such a way that it fulfills its task for each of the finite duration stationary processes given the different noise characteristics of each.

The crucial factor with this method of forming a number of stationary processes from a single non-stationary process is the duration of the stationary blocks. It would be desirable to make the blocks as long as possible so that the instrument would not need frequent re-adjustment. On the other hand, the blocks should be of a short enough duration such that the stationarity assumption is valid. There are several tests available which may assist the designer in determining the maximum duration of the blocks in an empirical manner, but these could not be applied on-line as they demand an excessive number of computations. Furthermore, the stationarity tests are inconclusive and subjective in that it is difficult to decide what constitutes an appreciable change in the statistics of the process. Performing a large number of tests prior to defining the instrument should, however, give the designer an insight into what strategy to adopt with an adaptive system.

5.3 CONCLUDING REMARKS

The presence of non-stationarities in a signal causes fundamental problems to the instrument designer. The effects of non-stationarities range from a loss of information to the possible invalidation of a particular measurement technique and the unpredictability of instrument performance. The subjectivity of non-stationarities precludes any possibility of a general approach to such measurement problems with the consequence that ad hoc solutions are often the only methods available to the designer. A better solution is to reduce a single non-stationary problem to a series of stationary ones and form an adaptive measurement system, but this requires extensive analysis prior to

instrument design; far more than would be required for a single stationary problem. Even with this method the subjectivity in the nature of non-stationarities creates problems in forming the stationary processes to a point where, more often than not, the designer is reduced to crude trial and error methods.

6. THE PROBLEM OF THE EPILEPTIC PRECURSOR AS AN EVENT

6.1. INTRODUCTION

An EEG record consists of a mapping of the electrical activity of the brain measured at up to 16 selected sites by electrodes placed on the subject's scalp. With the EEG record there may be a number of abnormalities which are of clinical importance, many of which occur at only infrequent intervals. In order to detect the occurence, therefore, of one of these abnormalities it may be necessary to examine 16 channels of information taken over a large period of time, a task to which the human observer is not well suited owing to the problems introduced by tedium, and one which consumes much of the valuable time of trained electroencephalographers. It is not surprising that attempts have been made to use a computer to detect these abnormalities, either as an end in itself or to reduce the original EEG record to a shorter one containing only the periods of interest (18).

The automatic detection of these abnormalities is not a simple task as the EEG signal, which is considered as background noise, exhibits several of the more difficult characteristics that must be considered when attempting to eliminate the noise component from a signal, such as non-stationarity and the intermittent presence of large amplitude noise know as artefact. The abnormalities may be either slow waves, sharp waves or spikes, giving a frequency range of interest of between 2 and 70 Hz; consequentially it is possible to employ a powerful microprocessor system to detect the abnormalities.

The abnormality of interest here is the spike-and-wave discharge which is indiciative of the onset of an epileptic attack. Often the presence of the spike alone is considered sufficient to indicate the onset of a epileptic attack, and indeed the spike detectors developed are among the most effective, and simplest, methods used to detect the epileptic precursor (4,19). A few of the more common techniques used to detect the spike-and-wave complex are discussed in Section 6.4, and analyses of the EEG and the spike-and-wave are given in Sections 6.2 and 6.3 respectively.

6.2. THE EEG

As stated above the EEG is derived from electrodes placed on the scalp of the subject. This provides a signal of only a few millivolts which is occasionally swamped by noise caused by muscular activity, such as eye movement, which is known as artefact. A signal of better quality may be obtained by implanted electrodes, and this is known as an electrocorticogram (ECoG), but as this is less convenient to obtain than the scalp electrode derived EEG it will be assumed that it is the EEG which is available for analysis. The EEG is obtained from up to 16 sites, and an abnormality may manifest itself at any one or more of these sites. The presence or absence of an abnormality at a number of sites and the delay between the time of arrival of the abnormality at these sites may be used to estimate the area of the brain in which the abnormality originated.

In a normal adult the EEG contains frequency components from about 6Hz to 30Hz, and in children frequencies as low as 2 or 3Hz are possible. The spectrum is not flat over this range with, in particular, a strong 10Hz component known as alpha activity (20). The abnormalities which may be present may either be slow waves in the 2 to 4Hz frequency range or sharp waves or spikes in the range 30 to 70Hz. Muscle artefact manifests itself as a burst of high amplitude noise in the same frequency range as the sharp waves and spikes.

The intermittent presence of artefact and abnormalities in the EEG implies that the statistics of the EEG vary with time; that is to say the signal is non-stationary. Even if these aberrations were not present the normal EEG varies depending on the state of rest of the subject (20,21). Thus any assumptions of stationarity are invalid. However, it is possible to divide a non-stationary signal into finite length records which are of sufficiently short duration such that each individual record may be considered as part of a stationary signal (see Chapter 5). Various estimates of the time over which the EEG may be considered stationary are available, due mainly to the subjectivity of the concept of stationarity, with estimates as short as 1 second proposed (22). The non-stationary aspect of the EEG is demonstrated in figure 6.1 where autocorrelograms derived from the EEGs of two subject are shown. The EEG records of each subject were divided into two blocks and the autocorrelation procedure was performed on each of the four blocks, thereby allowing a comparison of the autocorrelograms of two different subjects and of the EEG of each subject at two different times. Strictly speaking an autocorrelation is an ensemble average, that is to say it is derived from the distributions of the

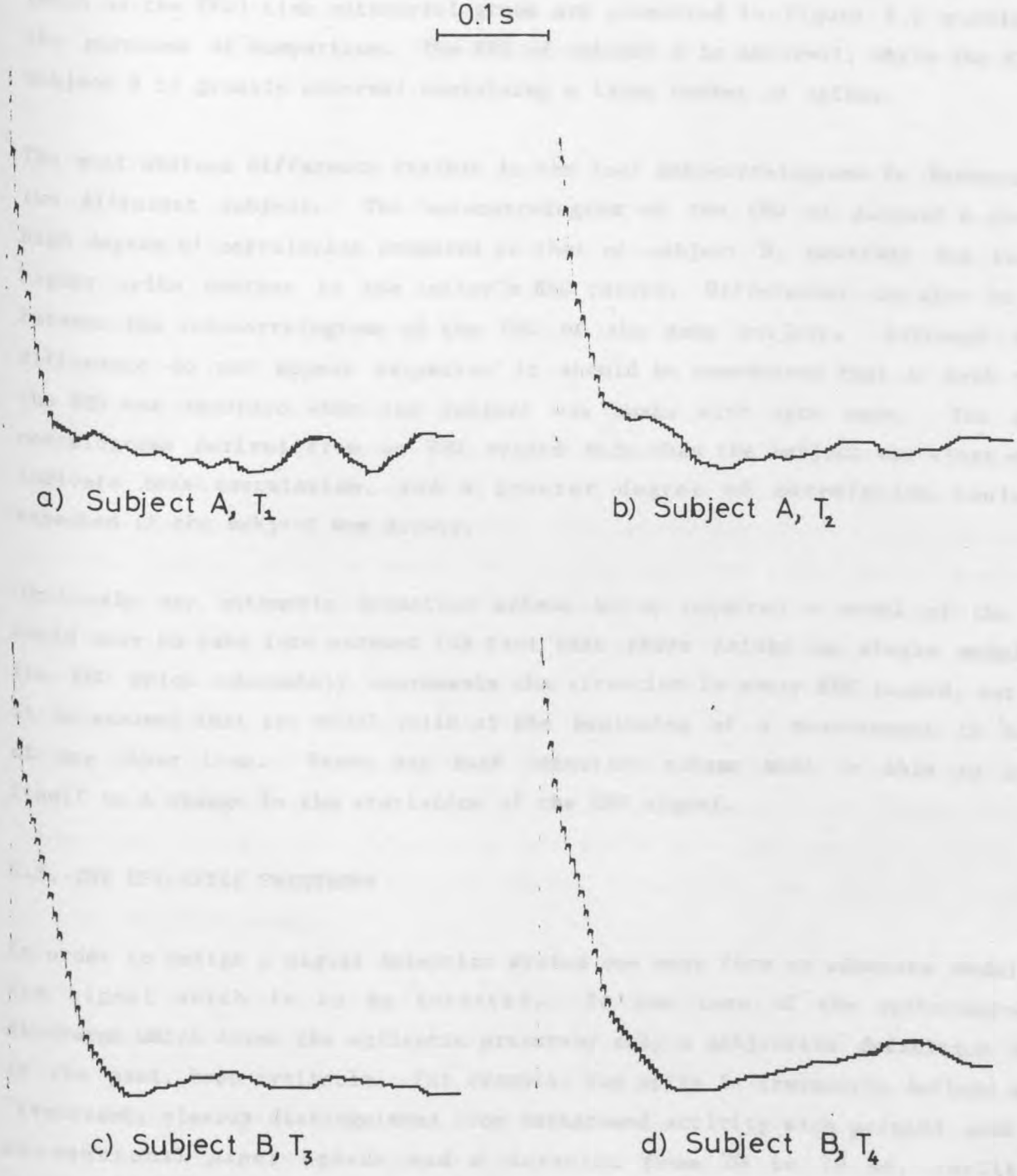


Figure 6.1 EEG Autocorrelograms

values of a family of functions (in this case the family of EEG records) at fixed times (16). It is rare that enough members of the family of functions are available for analysis in this way, so it is normal to make the assumption of ergodicity and replace the ensemble averages with time averages. Although it is not valid to make the ergodic assumption for non-stationary processes (such as the EEG) time autocorrelograms are presented in figure 6.1 purely for the purposes of comparison. The EEG of subject A is abnormal, while the EEG of subject B is grossly abnormal containing a large number of spikes.

The most obvious difference visible in the four autocorrelograms is between the two different subject. The autocorrelogram of the EEG of subject A shows a high degree of correlation compared to that of subject B, possibly due to the higher spike content in the latter's EEG record. Differences can also be seen between the autocorrelograms of the EEG of the same subject. Although these difference do not appear excessive it should be remembered that at both times the EEG was recorded when the subject was awake with eyes open. The autocorrelograms derived from an EEG record made when the subject was alert would indicate less correlation, and a greater degree of correlation would be expected if the subject was drowsy.

Obviously any automatic detection scheme which required a model of the EEG would have to take into account the fact that there exists no single model of the EEG which adequately represents the situation in every EEG record, nor can it be assumed that the model valid at the beginning of a measurement is valid at any other time. Hence any such detection scheme must be able to adapt itself to a change in the statistics of the EEG signal.

6.3. THE EPILEPTIC PRECURSOR

In order to design a signal detection system one must form an adequate model of the signal which is to be detected. In the case of the spike-and-wave discharge which forms the epileptic precursor only a subjective definition has, in the past, been available. For example, the spike is frequently defined as a "transient, clearly distinguished from background activity with pointed peak at conventional paper speeds and a duration from 20 to 70 mS, amplitude variable" (23). Slow waves are similarly defined as distinguishable features.

The above definition of the spike-and-wave is undesirable for several reasons. First and foremost is that the phrase "distinguished from background activity" is included for the benefit of the exceptional pattern recognition ability of

the human observer. However, it also allows a large degree of subjective interpretation by the human observer. Comley (4) states "the whole detection criterion used by human observers appears to be based more on an acquired, intuitive decision making process, rather than a formalised one". The pitfalls of such a scheme are exemplified in a experiment quoted by Brignell and Rhodes (ref: 1, Section 7.2) where 'skilled' operators measuring the breakdown strength of insulating oils would produce consistent - but nonetheless incorrect - results.

The effect of subjective human analysis in the detection of the epileptic precursor can be seen by an examination of the results quoted by Comley (4). In one experiment Comley ran four EEG records through a spike detection system in order to discover an optimum setting for the threshold detector. It was found that the optimum level was approximately 0.48 V for all but one record. On re-analysis, 11 of the 23 classified spikes on that record were found to be in fact sharp waves. Indeed, of the 117 spikes originally identified only 101 were found to be "correctly" identified (that is, a majority of observers deemed the features to be spikes). In addition the final spike detector system uncovered an extra 4 features which were later classified as spikes and 16 possible spikes, where a number of trained observers failed to agree upon the classification of the features. Other workers have also produced evidence that such an intuitive process may not be desirable. In one experiment (24) five trained electroencephalographers were asked to identify the sharp waves and spikes in identical EEG records. Of the features which were identified all five observers agreed on only 11% of them, while nearly half of the features were identified by only one observer. It is apparent that, among the other advantages offered, the objectivity available with an automatic detection scheme is essential if a rigorous method of detecting the epileptic precursor is considered desirable.

Returning to the definition of the epileptic precursor, one can see that there is not enough available information to create an adequate model of the spike-and-wave for a detection system. As it is essential for a good model to be constructed it was decided to ignore the standard definition of the spike-and-wave and generate a model directly from an EEG record.

The technique used to extract a model of the spike-and-wave is based on a method known as cumulative averaging (1). Normally cumulative averaging is used to extract the shape of a wave which is periodic, and as the occurence of the spike-and-wave is obviously not periodic a variation on the technique was used.

Consider a composite input signal, y(t), which is the sum of a periodic signal, p(t), and a noise component, say n(t), such that

$$y(t) = p(t) + n(t) \qquad 6.1$$

If the period of p(t) is T, say, then the shape of p(t) may be obtained by dividing y(t) into k blocks, each of duration T, and forming the ensemble average of these blocks, s(t)

$$s(t) = \frac{1}{k} \sum_{i=0}^{k-1} y(t + iT) , \quad 0 < t < T \qquad 6.2$$

Now, as p(t) is periodic then p(t) = p(t+T). Hence 6.2 reduces to

$$s(t) = p(t) + \frac{1}{k} \sum_{i=0}^{k-1} n(t + iT) , \quad 0 < t < T \qquad 6.3$$

Provided n(t) is not correlated with n(t+iT) it is progressively reduced as k increases. Thus the shape of p(t) may be extracted provided k is sufficiently large. As a rule, the noise component will be reduced in proportion to the square root of k.

In order to apply this averaging method to the problem of extracting the shape of the epileptic precursor from an EEG record is it necessary to detect and align the spike-and-waves, and it is fortunate that this has already been achieved by the spike detector produced by Comley (4). In this system an EEG record is passed via an 8-bit ADC to a CP1600 microcomputer system. The 8 bits of data are written to a digital magnetic cassette storage unit and when a

spike is detected the 9th bit available on the cassette is set. The cumulative averaging program (see Appendix III) uses this extra bit as a trigger to indicate the presence of a spike. This trigger bit is always set in the word corresponding to the sample immediately following the spike, and therefore the spikes are not only detected but also aligned.

However, as the trigger occurs after the spike it is necessary to store some pre-trigger information and this is achieved by using a cyclic buffer in which the latest sample obtained overwrites the oldest sample in the buffer. When a spike is detected both the pre- and post-trigger information are added into a cumulative average build-up data area, and a spike count incremented. When a run is completed, or when a sufficient number of spikes for the averaging routine are detected, the samples in the data area are divided by the number of spikes found.

The result of a run of the cumulative averaging program on a grossly abnormal EEG record is shown in figure 6.2. It can be seen that cumulative averaging is an extremely effective method of extracting a wave shape from a noisy signal.

Several points can be made about the derived spike-and-wave given in figure 6.2. The duration of the spike is approximately 36 mS, which is within the limits of 20 to 70 mS allowed for a spike in the standard definition given above. However, it is likely that the true spike will be of a slightly shorter duration, as the technique of cumulative averaging tends to stretch fast transients if there is any misalignment between the blocks over which the average is performed. In this case it is reasonable to expect that any misalignments will be of only one or two samples, or between 4 and 8 mS. The slow wave itself is of a definite shape, with a fast leading edge and a slower trailing edge. The total duration of the spike-and-wave is just over 0.3 S, which fits in quite well with the "three-a-second transient" definition of the epileptic precursor which is occasionally quoted. One particularly interesting feature is a distinct wave preceding the spike. It is, however, unlikely that this third part of the epileptic precursor would be of much value in an automatic detection scheme, as it is of a small amplitude and also a type of wave that one could reasonably expect to be present in a normal EEG.

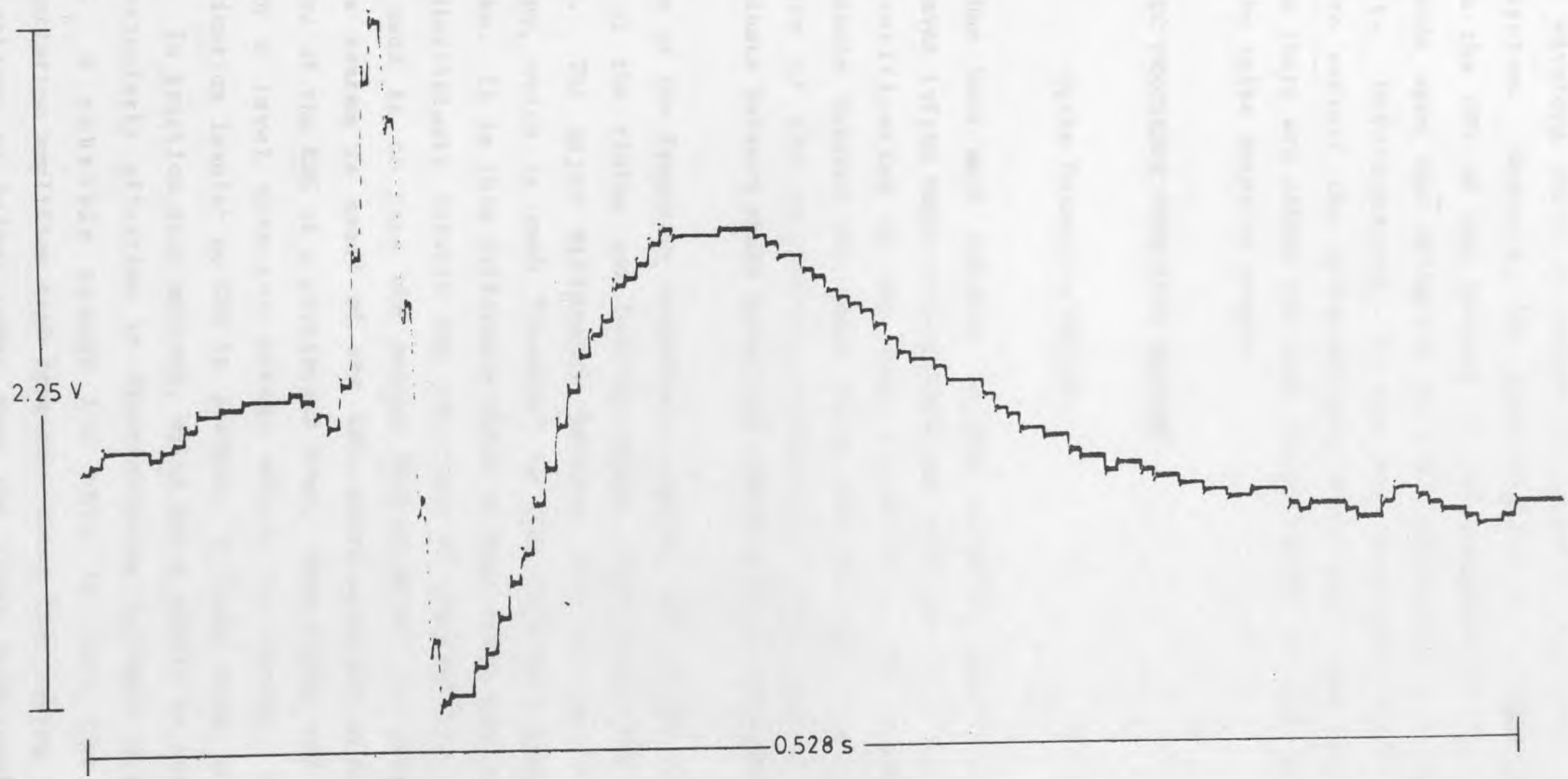


FIGURE 6.2 SPIKE-AND-WAVE EXTRACTED FROM AN EEG RECORD BY CUMULATIVE AVERAGING.

The modified method of cumulative averaging, then, can produce a well-defined sampled-data waveform which is emminently suitable for use in an automatic detection system. However, the applicability of a spike-and-wave complex derived from the EEG of one subject to the analysis of the EEG of another subject depends upon the variation in the spike-and-wave complex between the two subjects. Unfortunately, it has not been possible, with the data available, to extract the spike-and-wave discharges from records other than that used, as there are either too many false alarms or too few abnormalities present in the spike detector output.

6.4. EPILEPTIC PRECURSOR DETECTION METHODS

6.4.1. Spike Detection Methods

There has been much interest in the automatic detection of spikes and sharp waves (often known collectively as SSW) in EEG records, either for the classification of any fast transient or for further analysis to discriminate between the sharp waves and the spikes which indicate the presence of the epileptic precursor. Only those methods which discriminate between sharp waves and spikes will be considered here.

In terms of the frequency components present, and hence factors like the slopes of the rising and falling edges, sharp waves and spikes are very similar. The major difference between the two is the apex of the transient, which is much "rounder" in the sharp wave compared to that of the spike. It is this difference which is most often exploited in systems which discriminate between the two types of abnormality. The most common method used is to take the second differential (or difference, if a discrete system is used) of the EEG, which provided an indication of the curvature of the EEG at a particular time. This signal may then be passed through a level detector scheme which, in theory, may have three classification levels; no SSW is present, a sharp wave is present, or a spike. In practice such methods, while quite useful in detecting SSW, are not particularly effective in discriminating between spikes and sharp waves. A possible reason for this is that the operation of differentiation amplifies high frequency noise (1), making the calculation of derivatives of higher order than the first undesirable, particularly when the frequency range of interest is comparatively high, as it is with spikes and sharp waves.

A more effective method of spike detection has been developed independently by Comley (4) and Vera and Blume (19). The method used requires the first differential of the EEG to give an indication of the slope. If a fast positive going slope is followed by a fast negative going slope then a SSW is present. Discrimination between spikes and sharp waves is achieved by measuring the delay between these two slopes. The results quoted indicate success rates of better than 90% for "obvious" spikes in the EEG, and Vera and Blume quote similar figures for non-obvious spikes if the information is derived from an ECoG. This method of spike detection provides an acceptable success rate, and at the same time is simple enough for implementation on a microprocessor system.

All spike detection schemes suffer from the same disadvantage in that muscle artefact manifests itself as bursts of high frequency spikes, and when this is present a high number of false alarms can be expected. Several ways around this problem have been proposed. Comley suggests keeping a zero-crossing count, which should increase if artefact is present, but this could only indicate that the results given during the periods when the count is high are unreliable and does not assist in producing a valid result if artefact is present. Removing the artefact by, for example, the use of correlation techniques (25) would help, but such methods have met with, so far, little success. More promising solutions are based on two-tier systems where the presence of the epileptic precursor is indicated only if the spike is subsequently followed by a slow wave (26). As the slow wave is at the opposite end of the EEG spectrum from the muscle artefact it would be easier to distinguish than the spike. Any attempt to remove artefact, on the other hand, must to some extent attenuate the similar looking spikes.

6.4.2. Frequency Domain Methods

The epileptic precursor consists of two elements which have frequency components outside the normal range of the EEG. It is not, therefore, surprising that many automatic detection methods based on an analysis of the frequency content of the EEG have been tried. Theroretically, a band-stop filter which would pass frequencies below 5Hz or above 30 Hz would eliminate the normal EEG and pass any SSWs or slow waves which are present. Unfortunately the reality of the problem is less simple. At the higher end of the spectrum such a filter would also pass muscle artefact, resulting in a high false alarm rate. At the lower end of the spectrum a

filter with an extremely sharp cut-off would be required which, at such low frequencies, would be difficult to implement by continuous methods and, due to the sharp cut-off required, would require much processing time if implemented as a digital filter. There is also a problem of "ringing" with filters that have extremely sharp cut-offs; a direct result of the fact that the Fourier transform of a block function is of the form (sin x)/x, implying that a filter with an infinitely sharp cut-off has an infinite duration impulse response. Furthermore, the filter would have to be tailored to suit individual subjects whose EEG normally contains frequencies above or below those usually expected.

A novel method of constructing an individually tailored band-stop filter is the matched inverse digital filter (27). This system forms the spectral density of the normal EEG of an individual subject, inverts it and takes the inverse Fourier transform to provide an impulse response model of the filter which passes only those frequency components which are not contained in the normal EEG or which are present in greater power than in the normal EEG. Such a system would, of course, pass <u>any</u> non-stationarities and therefore the problem of artefact remains. In addition, as the EEG is itself non-stationary, the filter would have to be updated if, for example, the state of rest of the subject changes. The major problem with this system is that the EEG record from which the spectrum is derived must not contain any abnormalities which it is desired to detect, as such an abnormality would be considered as part of the normal EEG, and the resulting filter would reject the abnormality should it occur again.

In general, detection methods based on the frequency analysis of the EEG have met with little success, particularly when the results are compared with those of the spike detection methods given above. This lack of success does not seem to deter some workers, possibly because the abnormalities in the EEG are obviously outside the frequency range of the normal EEG, and so working in the frequency domain seems an obvious solution. However, it is possible that the technique of matched inverse filtering might be of some use in other applications where the background noise in a signal is stationary.

6.4.3. Pattern Recognition (Template Matching)

The problem of the detection of a known signal in noise is not uncommon, and there are a number of techniques available as a solution to this problem. Where the signal is comparatively complex pattern recognition techniques are often employed. Two such methods, which are in fact very similar if the noise component may be considered as white, are the correlation receiver and the matched filter (17). With a correlation receiver a "template" of the signal to be detected is cross-correlated with the incoming noisy signal, and if the output of the receiver exceeds a preset threshold then a signal is deemed to be present. A matched filter is one where the signal-to-noise ratio is maximised for a given signal in known noise. If the noise may be considered white then the impulse response of the matched filter is equivalent to the time-reversed version of the signal to be detected, hence the similarity between the matched filter and the correlation receiver.

The correlation receiver performs best when the background noise is uncorrelated, and as this is not the case with the EEG only the matched filter will be considered here. The matched filter, signal and noise are related by

$$\int_{o}^{\tau} h_o(z)\, R_{nn}(\tau - z)\, dz = s(T-\tau),\quad 0 \leqslant \tau \leqslant T \qquad 6.6$$

where h_o is the impulse response of the filter, R_{nn} is the auto-correlation of the background noise and s is the signal which is to be detected. Note that there appears to be a dimensional imbalance in equation 6.6. This arises from the autocorrelation being considered as a signal in its own right, rather than the average of the power of the background noise. Attempts have been made to apply the matched filter to the problem of the epileptic precursor, but these have met with little success. However, it is possible that the failure of the matched filter is due mainly to an inadequate analysis of the problem rather than the inapplicability of the technique.

There exists a sub-set of the general matched filter given above known as the simple matched filter. This makes the assumption that the background noise is white, that is to say the spectral density is a constant, often referred to as $N_o/2$. In this case the autocorrelation of the noise is given by

$$R_{nn}(\tau) = \frac{2}{N_o}\delta(\tau) \qquad 6.7$$

where $\delta(\tau)$ is the unit impulse function. Substituting this into equation 6.6. yields

$$-\frac{2}{N_o}\int_o^T h_o(z)\delta(\tau-z)dz = s(T-\tau),\ 0 \leqslant \tau \leqslant T \qquad 6.8$$

which may be shown to be

$$-\frac{2}{N_o}h_o(\tau) = s(T-\tau) \qquad 6.9$$

The constant $2/N_o$ makes no contribution to the signal-to-noise ratio and, therefore, it may be set to any convenient value. Hence for the simple matched filter the impulse response is proportional to the time-reversed version of the signal to be detected. The simple matched filter may also be applied if the white noise approximation can be made, i.e. if the noise is "white" over a bandwidth much larger than the bandwidth of the signal.

It is the simple matched filter which has previously been applied to the problem of the epileptic precursor and it can be seen from the discussion of the EEG in section 6.2 that a possible reason for its failure is the incorrect assumption of white noise. Indeed, not only is the EEG non-white, but the spike-and-wave has greater bandwidth than the normal EEG; obviously no assumption of whiteness can be made given this condition.

A further problem with the matched filter is due to the non-stationary nature of the EEG signal. It can be seen from equation 6.6 that the impulse response of the filter is dependant upon the autocorrelation of the background noise and this, as shown in section 6.2, is time-varying.

It is impossible, therefore, to design a single matched filter which will be optimal at all times. In particular, the autocorrelation of the EEG varies greatly between different subjects (as the spike-and-wave may do as well) and thus a different filter would need to be constructed for each subject.

6.5 ALTERNATIVE STRATEGIES

The high false alarm rates obtained by the use of the spike detector or frequency domain methods of indicating the presence of the epileptic precursor should not occur if a pattern recognition technique is employed. However, pattern recognition techniques such as the matched filter become less effective if the noise is non-white or non-stationary, as is the case with the EEG. If such techniques are to be used then modifications are required to overcome these problems.

The first problem is the non-whiteness of the EEG signal. This problem arises in many other signal detection systems and one solution is to pre-whiten the signal before the detection process is employed (17). Essentially, pre-whitening considers non-white noise as white noise which has been passed through a filter. An estimate of this (non-existent) filter is obtained and used to construct its inverse filter. If the original non-white signal is passed through the inverse filter then the output will have the properties of white noise, enabling the use of a simple matched filter or correlation receiver to detect the presence of a known signal. This is shown diagrammatically in figure 6.3. The drawback with the pre-whitening system is that the whitening filter will also act upon the signal as well as the noise and so the matched filter or correlation receiver would require modification to handle the change in the shape of the signal which is to be detected.

A better solution to the problem is to use the general form of the matched filter where the background noise is permitted to be non-white. From equation 6.6 it can be seen that, if the signal to be detected is known and the autocorrelation of the noise may be calculated, it is possible to derive the impulse response of the matched filter. The derivation of the impulse response of the filter could be performed by a microprocessor system which would also be used as the epileptic precursor detector, thereby allowing the generation of a filter which is matched to an individual subject's EEG. Furthermore, if the noise is non-stationary, as with the EEG, then the derivation of the filter

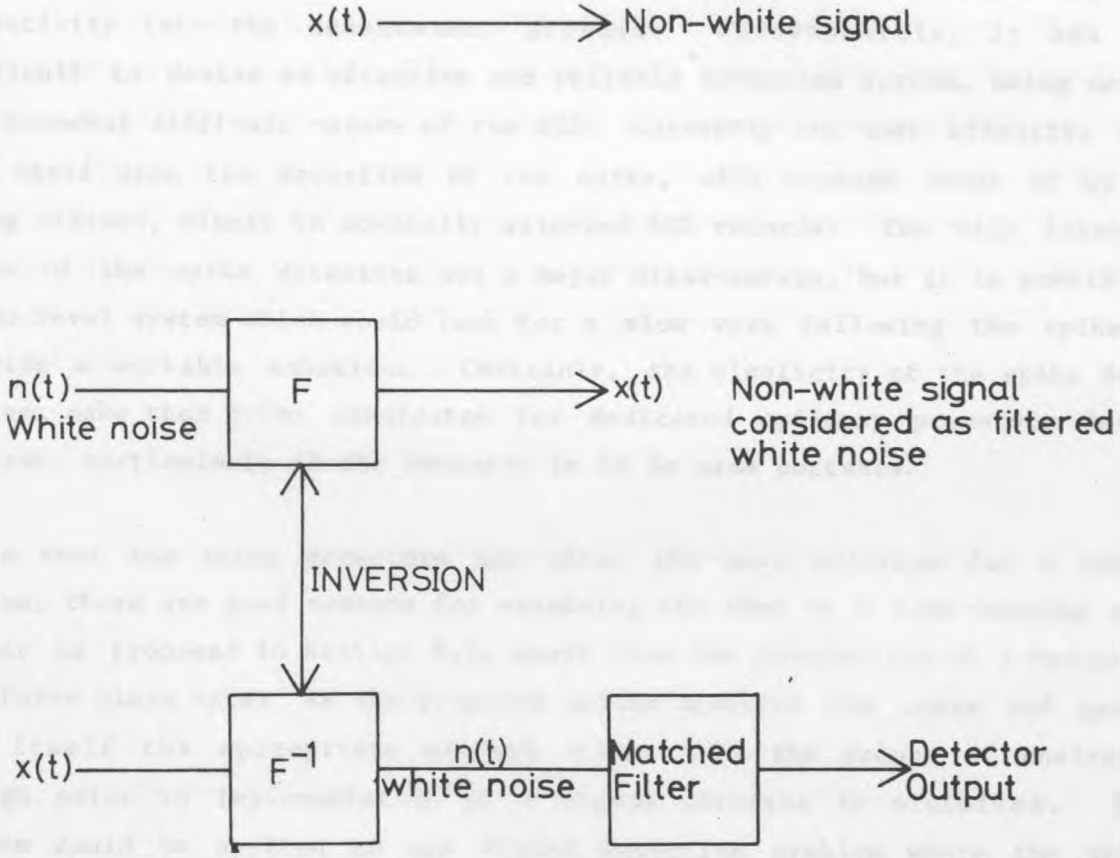


FIGURE 6.3 THE USE OF PRE-WHITENING IN SIGNAL DETECTION SYSTEMS

could be repeated during the course of a measurement, producing a series of filters which are individually matched to the varying background noise at particular times.

6.6 CONCLUSIONS

The automatic detection of the epileptic precursor is desirable for several reasons, not the least of which are the removal from human operators the tedious task of analysing large amounts of data and the introduction of some objectivity into the measurement process. Unfortunately, it has proved difficult to devise an effective and reliable detection system, owing mainly to the somewhat difficult nature of the EEG. Currently the most effective systems are based upon the detection of the spike, with success rates of up to 96% being claimed, albeit in specially selected EEG records. The high false alarm rates of the spike detectors are a major disadvantage, but it is possible that a two level system which would look for a slow wave following the spike would provide a workable solution. Certainly, the simplicity of the spike detector systems make them prime candidates for dedicated epilepsy precursor detection systems, particularly if the detector is to be made portable.

Given that the spike detectors may offer the best solution for a dedicated system, there are good reasons for examining the idea of a time-varying matched filter as proposed in section 6.5, apart from the possibility of a reduction in the false alarm rate. As the proposed system measures the noise and generates for itself the appropriate matched filter then the amount of analysis and design prior to implementation of a signal detector is minimised. Such a system could be applied to any signal detection problem where the noise is either unknown or time varying (within, of course, any constraints due to processing time, etc.). Such an instrument would be a powerful weapon in the CAM armoury, and the idea is expanded in the following chapter. Although it may not be the best solution for the epileptic precursor detection problem (nor the "best" for any such problem), its generality is of particular value and it is used on the EEG problem for the purposes of testing the concept.

7. POSSIBLE APPROACHES TO THE PROBLEM

7.1 INTRODUCTION

In Chapter 6 it was established that the matched filter which has the optimum signal-to-noise ratio for a given signal in noise satisfies the condition

$$\int_o^T h_o(z)\, R_{nn}(\tau - z)dz = s(T-\tau),\ 0 \leqslant \tau \leqslant T \qquad 7.1$$

Thus if the signal is given and the autocorrelation of the noise may be calculated then it should be possible to derive the filter coefficients. Unfortunately this requires finding the solution to an integral equation, a task to which a digital computer of limited power is not well suited. If an automatic method of determining the filter coefficient is to be found a further examination of the condition given in equation 7.1 is necessary. In particular, the similarity between this condition and the convolution operation is examined in the following section.

7.2 CONVOLUTION AND THE MATCHED FILTER CONDITION

The output $y(t)$ of a linear system with an impulse response $g(t)$ given an input $x(t)$ is given by

$$y(t) = g(t) * x(t) = \int_o^\infty g(\tau_1)\, x(t-\tau_1)d\tau_1 \qquad 7.2$$

where * is the convolution operator and τ_1 is a dummy variable (1).

Whalen (17) asserts, correctly, that the matched filter condition as given in equation 7.1 is not a convolution integral as the limits of integration are not the same. Consider, however, the case of a linear system with a finite duration impulse response, say $g_f(t)$. We may write $g_f(t)$ as a function of an infinite duration impulse response, say $g_i(t)$, such that

$$g_f(t) = g_i(t)[U(t) - U(t-T)] \qquad 7.3$$

where $U(t) = 0,\ t < 0$

$= 1,\ t > 0$

Thus the output y(t) of a linear system with a finite duration impulse response is given by

$$y(t) = \int_o^\infty g_i(\tau_1)\ [U(\tau_1)-U(\tau_1-T)]\ x(t-\tau_1)\ d\tau_1$$

$$= \int_o^\infty g_i(\tau_1)U(\tau_1)x(t-\tau_1)d\tau_1 - \int_o^\infty g_i(\tau_1)U(\tau_1-T)x(t-\tau_1)d\tau_1 \qquad 7.4$$

Now, for $\tau_1 \geqslant 0$, $U(\tau_1) = 1$. Therefore,

$$y(t) = \int_o^\infty g_i(\tau_1)x(t-\tau_1)d\tau_1 - \int_o^\infty g_i(\tau_1)U(\tau_1-T)x(t-\tau_1)d\tau_1 \qquad 7.5$$

Furthermore, for $\tau_1 < T$, $U(\tau_1-T) = 0$ and for $\tau_1 > T$ $U(\tau_1-T) = 1$. Thus we may write

$$y(t) = \int_o^\infty g_i(\tau_1)x(t-\tau_1)d\tau_1 - \int_T^\infty g_i(\tau_1)x(t-\tau_1)d\tau_1$$

$$= \int_o^T g_i(\tau_1)x(t-\tau_1)d\tau_1 \qquad 7.6$$

Comparing this result with the matched filter condition given in equation 7.1 we can see that, provided the condition that $0 \leqslant t \leqslant T$ is met, the matched filter may be considered as a linear system with a finite duration impulse response which, when the autocorrelation of the noise is applied as the input, gives as its output the time reversed version of the signal to be detected. This offers several techniques for deriving the coefficient of the matched filter.

7.3 METHODS OF DERIVING THE IMPULSE RESPONSE OF A LINEAR SYSTEM

As shown above we may consider the matched filter as a linear system with a finite duration impulse response, of which the input and output are known, but the system parameters are, as yet, undetermined. There are a number of techniques for deriving the parameters of such a system and these may be divided into three categories; direct methods (deconvolution), optimisation methods and system identification techniques.

7.3.1 Direct Methods

In section 7.2 it was shown that the performance of a linear system is described by the operation of convolution

$$y(t) = g(t) * x(t) \qquad 7.7$$

Thus it is necessary to perform a deconvolution operation to determine the impulse response of the system given the input and output. In the time domain the operation of deconvolution is practically impossible and it is usual to make a transformation to the frequency domain via the Fourier transform. The operations of convolution and multiplication form a Fourier transform pair and thus transforming equation 7.7 we get

$$Y(\omega) = G(\omega)X(\omega) \qquad 7.8$$

It is now relatively simple to extract the Fourier transform of the impulse response (transfer function)

$$G(\omega) = \frac{Y(\omega)}{X(\omega)} \qquad 7.9$$

In the case of the matched filter this method would require the Fourier transforms of the time reversed version of the signal to be detected and the autocorrelation (the power spectrum). As the signal to be detected is assumed to be time-invariant it may be supplied as a frequency domain model, but the power spectrum must still be calculated.

Once the Fourier transform of the impulse response of the matched filter is known there exists two methods of implementing the filter in order to perform the required measurement. One method is to take the inverse Fourier transform of the transfer function and convolve this with the incoming signal to give the filtered output. Alternatively, the Fourier transform of the incoming signal could be multiplied by the transfer function to give the frequency domain version of the output. The inverse Fourier transform of the output would be required, as the interpretation of the frequency domain version of the output of the matched filter would be a problem. The first implementation is better from a practical viewpoint as it requires one less transformation.

The advantage of the above method of deriving the coefficient of the matched filter is that it is comparatively straightforward and provides the impulse response of the filter directly. The disadvantage is that this is achieved at the cost of a large amount of processing and, therefore, it is only feasible in high performance systems (where, for example, the Fourier transforms are performed at least partly in hardware). In particular, it is a suitable method if the noise is time-invariant, as the matched filter will be fixed for the duration of the measurement so that the coefficients need only be calculated once. However, for the application considered here where the noise is non-stationary and the calculations are performed on a portable microprocessor system, the direct method of deriving the filter coefficient is unacceptable.

7.3.2 Optimization Methods

In general, optimization methods are trial-and-error techniques used to find the maxima, minima or zero points of a function. A sequence of guesses are made to the solution, each guess being either retained or rejected according to whether it is better or worse in some sense to the previous guess. The differences between the various methods depends upon how the next guess in the sequence is generated and whether there exist any constraints on the range of possible values.

We may consider the problem of the derivation of the matched filter parameters as a bounded domain optimization problem (note that in optimization nomenclature a bounded problem and a constrained problem are not the same; there are no constraints in the matched filter coefficient derivation problem). This is because, for practical reasons, the range of values permitted are restricted to those which may be represented as a fixed point number. The function to be minimised is the error between the output of the matched filter when the autocorrelation of the noise is applied as the input and the time reversed version of the signal to be

detected. This error may be expressed as the square of the Euclidean distance between the ideal and the current estimate thus

$$J = \sum_{i=1}^{N} (S^*_i - S_i)^2 \qquad 7.10$$

where S_i is the time reversed version of the signal to the detected and S^*_i is the output of the current estimate of the matched filter. J is sometimes known as the performance index or function. As S^*_i is related to the changes in the filter coefficients by the convolution operation it should be calculated as infrequently as possible. This implies that the derivatives of the performance index are inaccessible so that gradient descent optimization methods cannot be employed and we must rely upon some form of direct search.

The derivation of the matched filter coefficients can, therefore, be formulated as an unconstrained direct search over a bounded domain. This is quite a common form of optimization problem and the literature on this subject is extensive. Reviews of unconstrained optimization methods are presented by Powell (28) and Fletcher (29) and a number of the techniques mentioned are explained by Pun (30). For the problem in hand, where the optimization program is to run on a computer of limited power, Nash (31) provides excellent background reading. A few of the available techniques are given below.

7.3.2.1 Random Search

As its name suggests the random search is little more than a crude trial-and-error method of determining an optimum condition. It consists of either generating pseudo-random numbers and using these as values in the function or of generating a pseudo-random vector to give a displacement from the current estimate. In all but the most ill-behaved functions the vector displacement method is preferred, especially if a good initial estimate can be provided. The most methodical way of implementing a random search is to generate new estimates in the function space immediately around the current estimate and gradually extend the range of the search with each successive failure. This may be done either by combining a randomly generated vector direction with a step length which may be increased

or by generating random numbers with a suitable probability density function, the dispersion of which may be easily increased. For the latter method generating random numbers according to the Laplace distribution, which has a probability density function of

$$f(x) = ae^{-x} \quad , \quad x \geqslant 0$$
$$= ae^{x} \quad , \quad x < 0 \qquad 7.11$$

is suitable, as it may be generated from a uniform distribution quite simply by analytic inversion (32) while at the same time having a single dispersion parameter.

The disadvantages of the random search arise mainly because it takes no account of the behaviour of the function. As a consequence one would expect a large number of unsuccessful estimates to be generated and as in this case each estimate requires the calculation of the convolution of the estimate of the filter and the autocorrelation of the noise the use of the random search to find the filter parameters is limited. Furthermore, the random search gives no indication that an optimum point has been found - only that one point is better or worse than another. This implies that the convergence of the random search is unpredictable; indeed, it does not converge in the usual sense. Having said that, it should be noted that the random search, unlike other optimization techniques, does not get trapped by local optima. Thus although it might be foolish to attempt to find the optimum of a function by a random search alone, it would be prudent to intersperse other more educated techniques with a random search if it is suspected that the function is multi-modal.

7.3.2.2. Grid Search

The grid search is a more methodical method of determining an optimum point of a function. It consists of dividing the function space into a coarse grid and taking one point from each section of the grid. The optimum of these points is found and the section in which it resides is itself divided into a grid and the procedure repeated. A graphical representation of this technique for the 2-dimensional case is given in figure 7.1. the diagram shows 3 steps in the process, where the range of each value is reduced by a factor

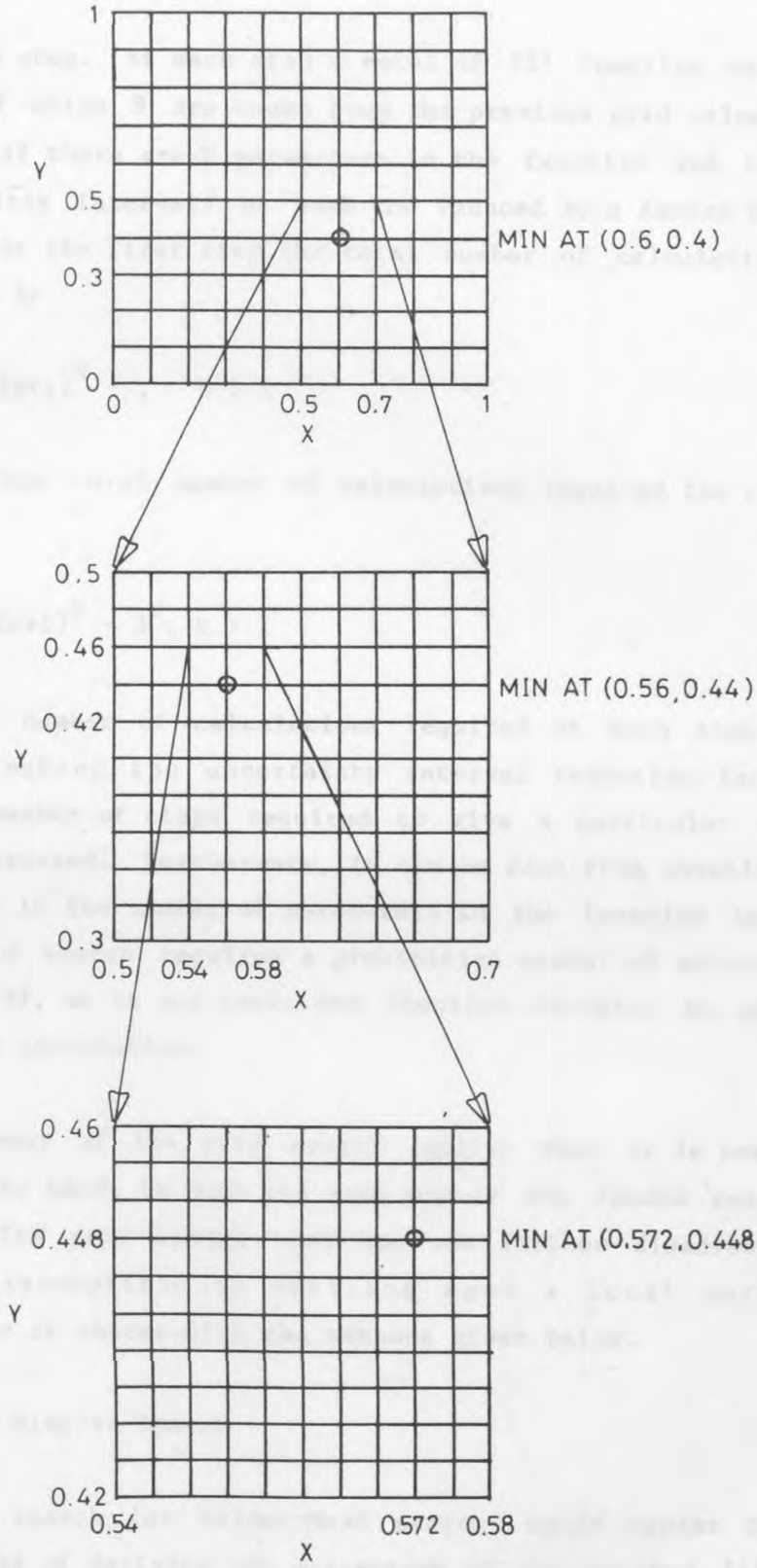


FIGURE 7.1 2-DIMENSIONAL GRID SEARCH

of 5 at each step. At each step a total of 121 function values are required, of which 9 are known from the previous grid calculations. In general, if there are N parameters in the function and the range (or uncertainty interval) of each is reduced by a factor k at each step, then for the first step the total number of calculations, say C_T, is given by

$$C_T = (2k+1)^N \quad , \quad k > 1 \qquad 7.12$$

Thereafter, the total number of calculations required for each step is given by

$$C_T = (2k+1)^N - 3^N, \; k > 1 \qquad 7.13$$

Although the number of calculations required at each step may be reduced by making the uncertainty interval reduction factor, k, smaller the number of steps required to give a particular accuracy would be increased. Furthermore, it can be seen from equations 7.12 and 7.13 that if the number of parameters in the function is large, then the grid search requires a prohibitive number of calculations, particularly if, as in our case, the function contains an operation as complex as convolution.

The inefficiency of the grid search implies that it is unsuitable for the task in hand, in much the same way as the random search was discarded. The grid search also has one further disadvantage in that it is susceptible to settling upon a local optima; a characteristic it shares with the methods given below.

7.3.2.3. The Simplex Search

The simplex search (or Nelder-Mead search) would appear to be a suitable method of deriving the parameters of the matched filter as it requires only the function values. It is more efficient than the grid search, although this gain in efficiency is obtained through a more complex algorithm and an increase in storage requirements.

A simplex, in n-dimensional space, is a solid having plane faces and n+1 vertices. Thus in 2 dimensions a triangle is a simplex (see figure 7.2) while a tetrahedron is a simplex in 3 dimensions. In

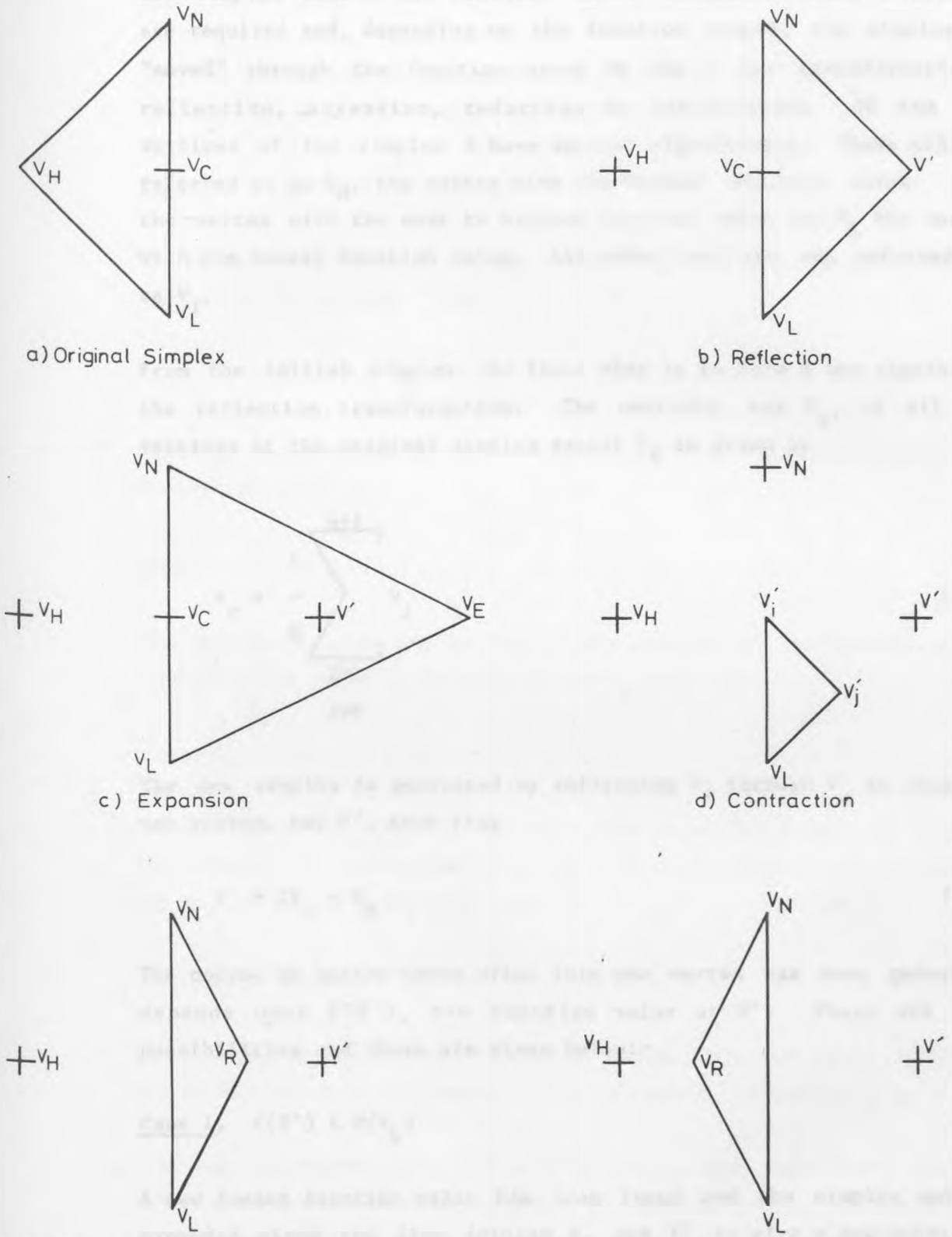


Figure 7.2 2-Dimensional Simplex and Transformations

the simplex search the function values at each of the n+1 vertices are required and, depending on the function values, the simplex is "moved" through the function space by one of four transformations; reflection, expansion, reduction or contraction. Of the n+1 vertices of the simplex 3 have special significance. These will be referred to as V_H, the vertex with the highest function value; V_N, the vertex with the next to highest function value and V_L the vertex with the lowest function value. All other vertices are referred to as V_i.

From the initial simplex the first step is to form a new simplex by the reflection transformation. The centroid, say V_c, of all the vertices of the original simplex except V_H is given by

$$V_c = \frac{1}{N} \sum_{\substack{i=1 \\ i \neq H}}^{n+1} V_i \qquad 7.14$$

The new simplex is generated by reflecting V_H through V_c to obtain a new vertex, say V', such that

$$V' = 2V_c - V_H \qquad 7.15$$

The course of action taken after this new vertex has been generated depends upon $F(V')$, the function value at V'. There are four possibilities and these are given below:-

Case 1. $F(V') < F(V_L)$

A new lowest function value has been found and the simplex may be expanded along the line joining V_C and V' to give a new point, V_E such that

$$V_E = 2V' - V_C \qquad 7.16$$

If $F(V_E) < F(V')$ then V_H is replaced by V_E, otherwise it is replaced by V'.

Case 2. $F(V_L) \leqslant F(V') < F(V_N)$

The new vertex is of no special significance and so V_H is simply replaced by V' with no other action being taken.

Case 3. $F(V_N) \leqslant F(V') < F(V_H)$

The simplex is reduced on the side of the reflection by finding a new vertex V_R between V' and V_C.

$$V_R = 0.5(V' + V_C) \qquad 7.17$$

If $F(V_R)$ is greater than both $F(V_H)$ and $F(V')$ then the simplex is contracted about V_L.

Case 4. $F(V') \geqslant F(V_H)$

The simplex is reduced on the side opposite to the reflection by finding a new point V_R between V_C and V_H such that

$$V_R = 0.5(V_H + V_C) \qquad 7.18$$

As with case 3 if $F(V_R)$ is greater than both $F(V_H)$ and $F(V')$ then the simplex is contracted about V_L. This is achieved by creating a set of new vertices V'_i such that

$$V_i' = 0.5(V_L + V_i)$$

The similarity between the courses of action taken in cases 3 and 4 may be exploited in an algorithm by, in case 3, replacing V_H by V'.

The simplex search has a number of disadvantages, especially when it is to be implemented on a computer of limited power. Firstly the storage requirement for intermediate results is very high if a large number of parameters are present. Using the algorithm presented by Nash (31) a total of (n+1) by (n+2) locations are required. For the derivation of the impulse response of the matched filter, where it is reasonable to expect up to 64 parameters, this would mean a storage requirement of 4290 words. In machines where the precision of the arithmetic is limited, a number of problems exist, for

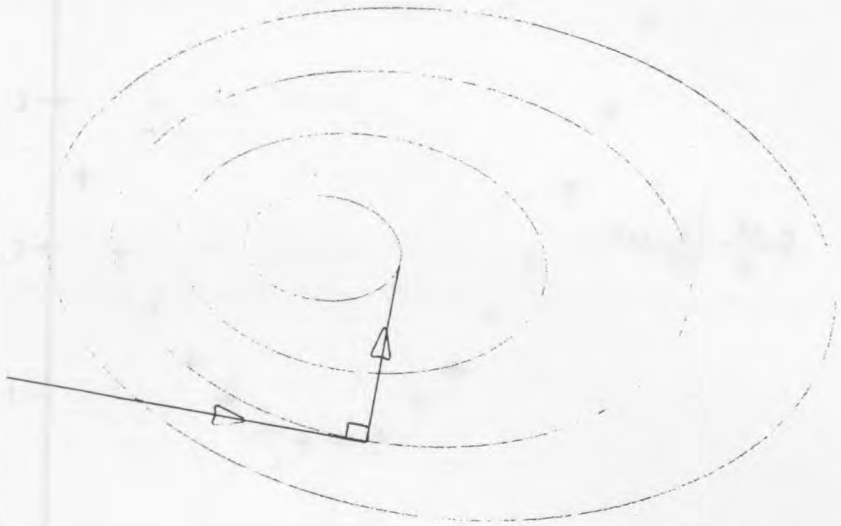
example a contraction may not actually reduce the size of the simplex, or all the vertices may exist in one plane implying that a move towards a minimum may not be possible (31). In addition, the simplex search becomes inefficient if the number of parameters becomes large (28, 29, 31), although it will be no less efficient than the grid search.

7.3.2.4. Vector Search

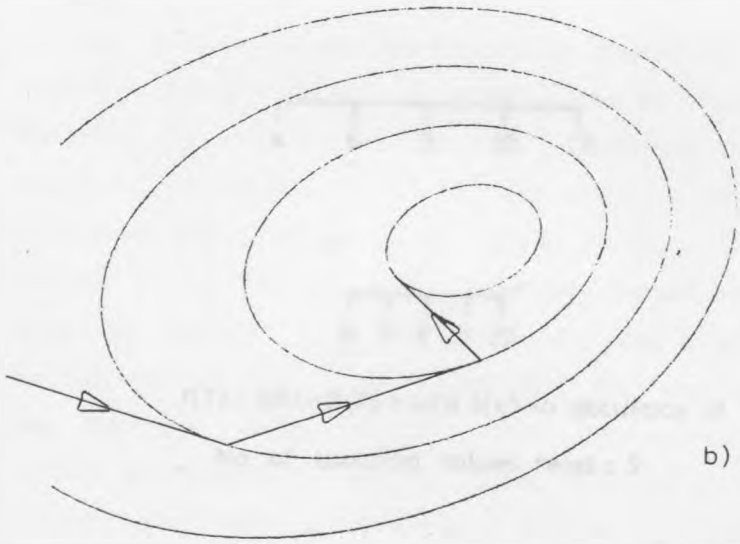
The vector search method of approaching an optimum consists of making steps along a vector until the optimal point on that vector is found. Once the optimal point is ascertained, another vector passing through this point is generated and a search is initiated along this new vector. The process continues until an acceptable approximation to the optimum is found. Two examples are shown diagrammatically in figure 7.3 for the two-dimensional case.

There are several methods available for obtaining the optimal point on a vector, but for the bounded domain problem the simplest technique is called binary sub-division (or dichotomous search). This is a one-dimension version of the grid search where the uncertainty interval reduction factor is 2. From equations 7.12 and 7.13 the number of function values initially required is 5 and, thereafter, 2 new function values are required at each step (see figure 7.4). A more efficient technique is the Fibonacci search which, although it may require more steps than the binary search (on average the number of steps will be about the same), requires only 1 new function value at each step. The search is based on the Fibonacci sequence, which is defined as

$$F_o = F_1 = 1$$
$$F_N = F_{N-1} + F_{N-2} \quad , N \geqslant 2 \qquad 7.20$$

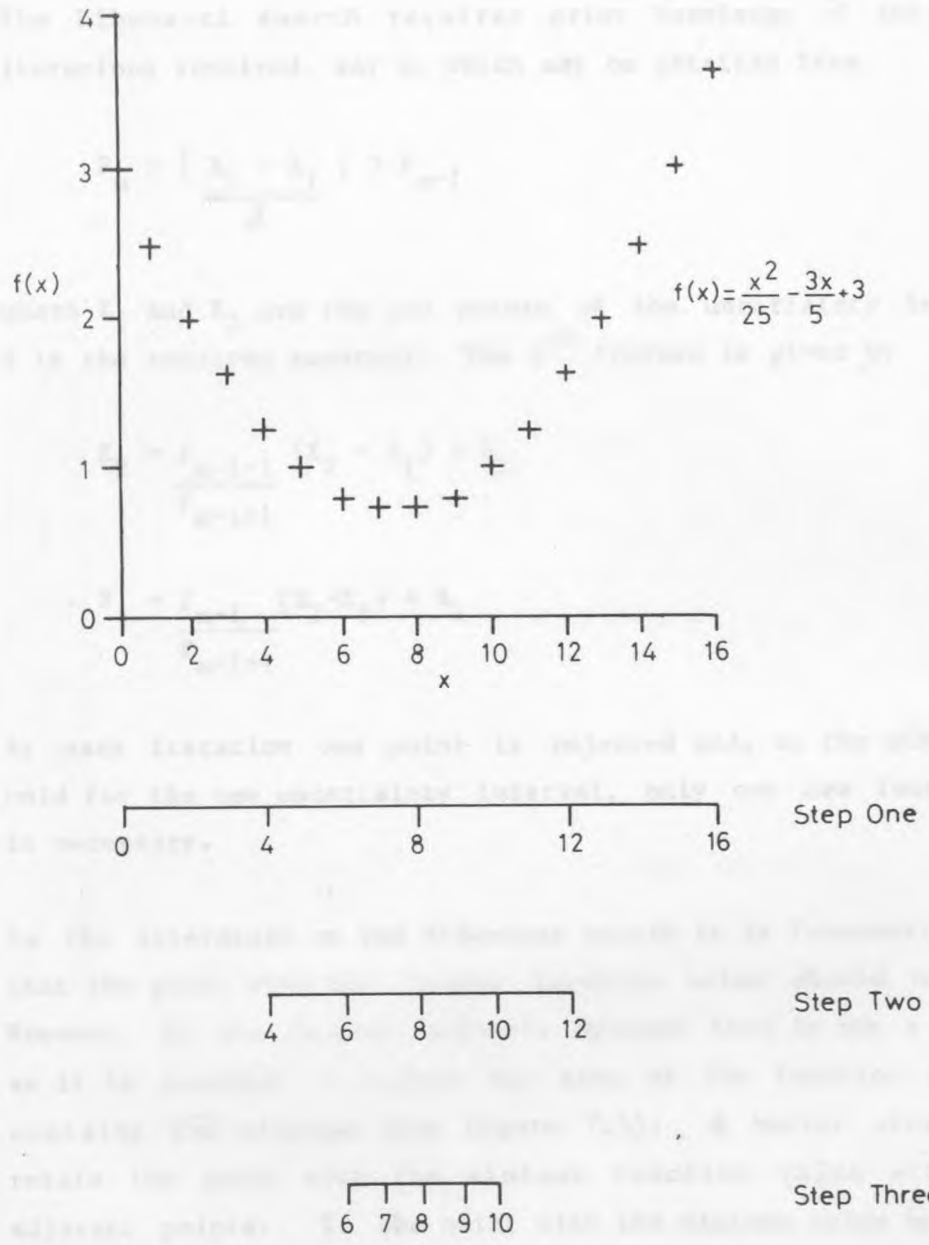


a) Orthogonal



b) Non-orthogonal

Figure 7.3 2-Dimensional Vector Searches



f(7) = f(8) = 0.76 = MIN f(x) to accuracy of 1.

No. of function values reqd.= 9

Figure 7.4 Binary Sub-division

The Fibonacci search requires prior knowledge of the number of iterations required, say m, which may be obtained from

$$F_m > \left| \frac{X_2 - X_1}{\delta} \right| > F_{m-1} \qquad 7.21$$

where X_1 and X_2 are the end points of the uncertainty interval and δ is the required accuracy. The i^{th} iterate is given by

$$X_2 = \frac{F_{m-i-1}}{F_{m-i+1}} (X_2 - X_1) + X_1$$

$$X_4 = \frac{F_{m-1}}{F_{m-i+1}} (X_2 - X_1) + X_1 \qquad 7.22$$

At each iteration one point is rejected and, as the other 3 points hold for the new uncertainty interval, only one new function value is necessary.

In the literature on the Fibonacci search it is frequently suggested that the point with the largest function value should be rejected. However, in the present author's opinion this is not a good policy as it is possible to reject the area of the function space which contains the minimum (see figure 7.5). A better strategy is to retain the point with the minimum function value with the two adjacent points. If the point with the minimum value happens to be an endpoint it should be saved with its one adjacent point and two new function values found. This may appear inefficient as two new values are required, but it is not because a complete iteration has effectively been eliminated. Figure 7.6 shows the Fibonacci search where the range is 21 and the accuracy required is 1. An alternative to the Fibonacci search is the Golden Section method. This uses the fact that, for large m

$$\frac{F_{m+1}}{F_m} \approx \frac{1}{2} (1+\sqrt{5}) = \tau \qquad 7.23$$

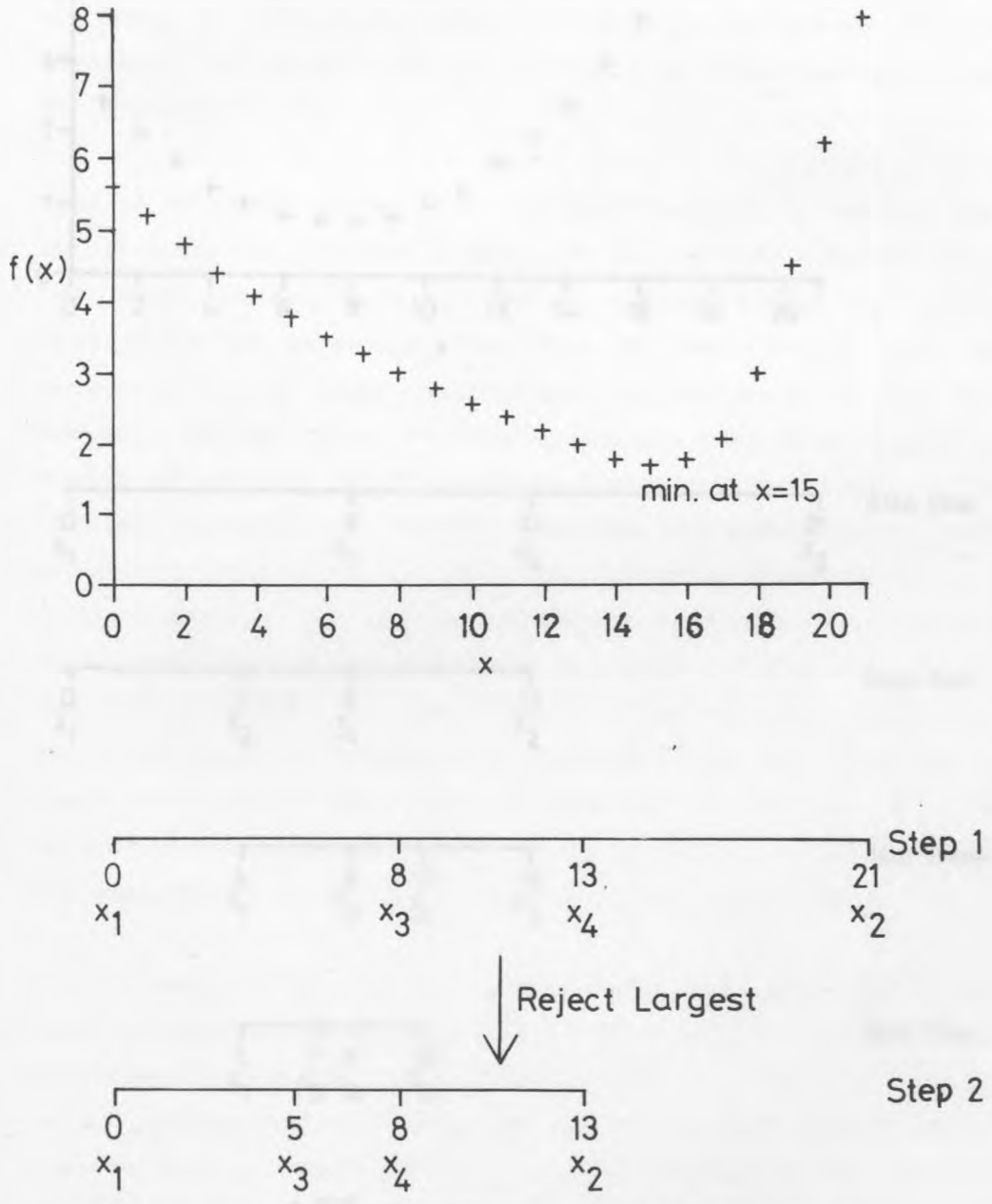


FIGURE 7.5 FAILURE OF FIBONACCI SEARCH IF LARGEST FUNCTION VALUE IS REJECTED

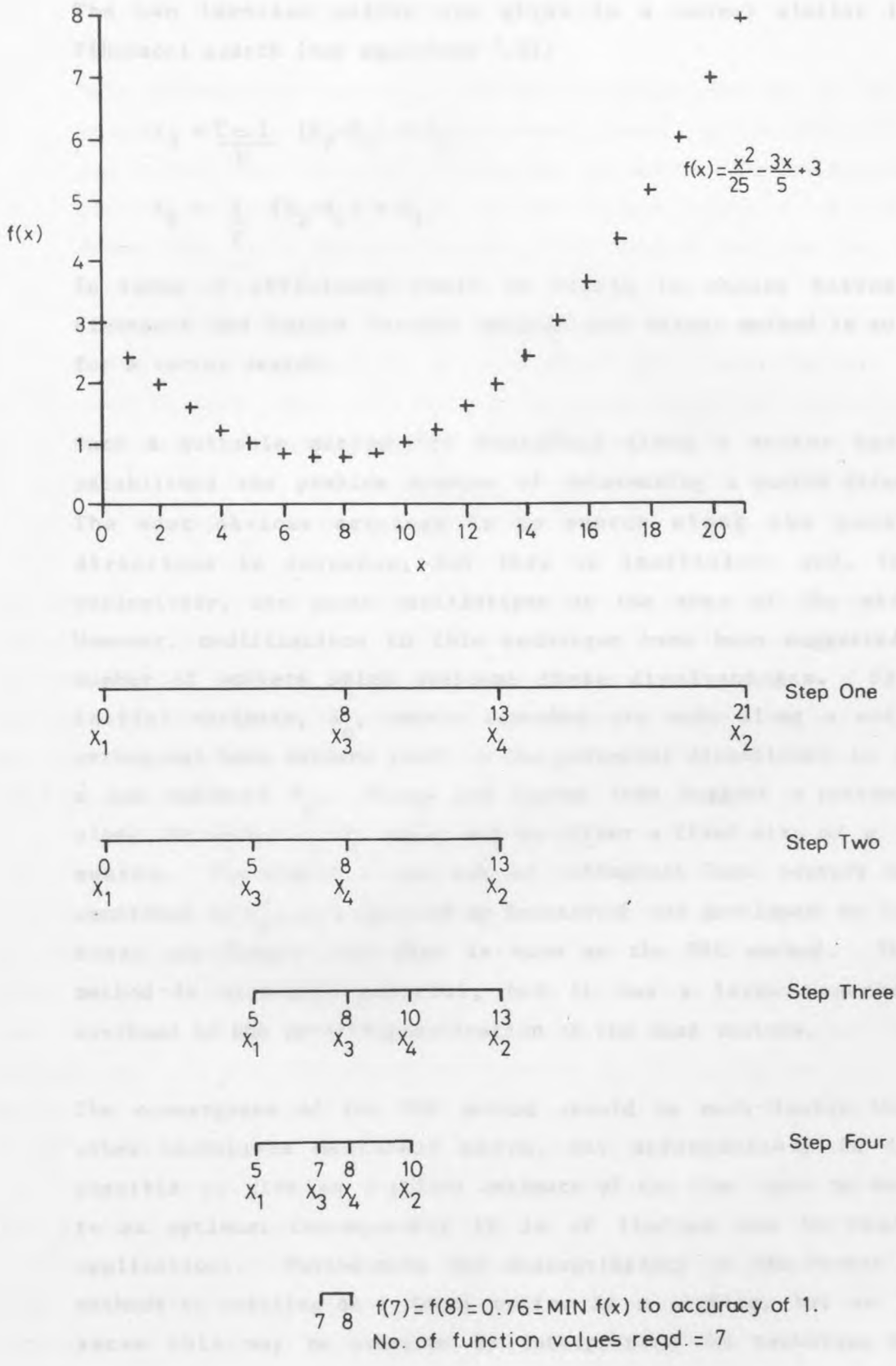


FIGURE 7.6 THE FIBONACCI SEARCH

The two interior points are given in a manner similar to the Fibonacci search (see equations 7.22)

$$X_3 = \frac{\tau - 1}{\tau} (X_2 - X_1) + X_1$$

$$X_4 = \frac{1}{\tau} (X_2 - X_1) + X_1 \qquad 7.24$$

In terms of efficiency there is little to choose between the Fibonacci and Golden Section methods and either method is suitable for a vector search.

Once a suitable method for searching along a vector has been established the problem remains of determining a search direction. The most obvious strategy is to search along the parameter directions in sequence, but this is inefficient and, if used exclusively, can cause oscillations in the area of the minimum. However, modifications to this technique have been suggested by a number of workers which overcome these disadvantages. From an initial estimate, X_o, vector searches are made along a set of n orthogonal base vectors (such as the parameter directions) to obtain a new estimate X_n. Hooke and Jeeves then suggest a pattern move along the vector $X_n - X_o$, which may be either a fixed step or a vector search. Conversely a new set of orthogonal base vectors may be generated at X_n, as suggested by Rosenbrock and developed by Davies, Swann and Campey into what is know as the DSC method. The DSC method is extremely powerful, but it has a large housekeeping overhead in the re-orthogonalization of the base vectors.

The convergence of the DSC method should be much faster than the other techniques mentioned above, but unfortunately it is not possible to give an a priori estimate of the time taken to converge to an optimum. Consequently it is of limited use in real-time applications. Furthermore the susceptibility of the vector search methods to settling on a local optima is a problem, but as stated above this may be overcome by interpreting the technique with a random search.

7.3.2.5 Comments On Optimization Methods

Most optimization methods have been invented for use on off-line computer systems with high processing power or for problems which are analytical. Although this makes the techniques unsuitable for implementation on a portable microprocessor system in a real-time environment it is conceivable that such methods are the only tools available to the programmer. If this is the case, then the DSC method would appear to be the likeliest solution. It should be noted that, as with the direct method of determining the filter coefficients, an optimization solution would be suitable for applications involving time-invariant noise.

7.3.3. System Identification Techniques

In order to control a system it is essential to discern the response of the system to a given input. The importance of system identification to control theory has led to the development of numerous methods of deriving the response of a given system. The techniques mentioned in this section are covered by Graupe (33) and only a brief description of these methods will be given here.

The identification technique used in a particular problem depends upon the amount of a priori knowledge of the system. In the case of the derivation of the coefficients of the matched filters it is fortunate that there is a great deal of a priori information. It is known that the system is linear and (for the period over which the identification is to take place) stationary. As the signal to be detected is defined by the user of the system the dimensionality of the parameter vector is known, as is the range of values which a particular parameter may possess.

The problem of deriving the coefficients of the matched filter differs from the classical control theory models in that the system does not exist when the identification procedure begins; only the input and output of the system are available. Thus techniques which require the application of special inputs (pseudo-random noise, steps, impulses, etc.) are not applicable.

7.3.3.1. Regression Techniques

Regression methods of system identification produce least-squares error estimates of the system parameters of a linear and stationary process. The derived model of the system may be in impulse response form. The techniques are, therefore, applicable to the problem of the derivation of the matched filter coefficients.

The technique used is to take a number of measurements of the input and output of the system to be identified. Provided the number of measurements exceed the number of parameters in the system the equations relating the input and output may be solved simultaneously. If the input matrix is represented by X, the output vector by Y and the parameter vector by G then the least-squares error estimate of the parameters is given by

$$\hat{G} = (X^T X)^{-1} X^T Y \qquad 7.25$$

where $\hat{G}$ is the best estimate of the system parameters.

As equation 7.25 requires the operation of matrix inversion it is not suitable for implementation in a real-time system. Fortunately, there exist sequential regression methods which do not require matrix inversion. Essentially, sequential regression generates a new estimate of the parameters of the system by adding an error correction vector to the present estimate thus

$$\hat{g}_n = \hat{g}_{n-1} + P_n x_n (y_n - x_n^T \hat{g}_{n-1}) \qquad 7.26$$

where $\hat{g}_n$ is the estimate of the parameters after n measurements and P_n is the step length used after n measurements. The calculation of P_n is complex and it is given by

$$P_n = P_{n-1} - P_{n-1} x_n (1 + x_n^T P_{n-1} x_n)^{-1} x_n^T P_{n-1} \qquad 7.27$$

This does not require a matrix inversion as $(1+x_n^T P_{n-1} x_n)$ is a scalar. The initial value, P_o, should be of the form

$$P_o = KI \qquad 7.28$$

where I is the identity matrix and K is the largest number which may be contained in the computer word.

Sequential regression is a possible method of extracting the matched filter parameters, but the calculation time required may still be too high for a satisfactory solution. However, the calculation time for the sequential regression identification method is far less than the time taken by the optimization methods given in section 7.3.2.

7.3.3.2. Sequential Learning Identification

Sequential learning identification is similar to sequential regression in that a new estimate of the system parameters is generated by adding an error-correcting vector to the present estimate. The calculation of the error-correcting vector is somewhat simpler with sequential learning identification, consequentially the rate of convergence is slower than that of the regression method. It does, however, converge to the true parameters rather than a least-squares error estimate. The model derived is in impulse response form.

Using the notation given in section 7.3.3.1 the n^{th} estimate of the impulse response of the system is given by

$$\hat{g}_n = \hat{g}_{n-1} + \alpha(y_{n-1} - \hat{y}_{n-1}) x_{n-1} / x_{n-1}^T x_{n-1} \qquad 7.29$$

where α is an error-correcting coefficient. For rapid convergence α should be less than unity and for a small steady-state error α should be much less than unity.

The sequential learning identification algorithm is much simpler than those for either regression techniques or optimization methods. It is, therefore, more suitable for implementation on a portable microprocessor system working in real-time. Other system identification techniques are either inapplicable or converge too

slowly to be useful. It is suggested that a method of determining the coefficients of the matched filter be based on the technique of sequential learning identification.

7.4 THE USE AND IDENTIFICATION OF THE MATCHED FILTER

At any one time throughout the course of a measurement there exists an approximation to the matched filter which is optimal for the signal to be detected in the measured noise. It is this approximation which is used to perform the actual measurement. As the measurement progresses the quality of the noise changes and a new approximation of the filter is required. Thus the identification algorithm and the filter program must either run concurrently or the identification procedure must be allocated some processing time, thereby reducing the system's real-time capability.

It should be noted that the autocorrelation of the composite signal, rather than just the noise, is used. This is valid provided the signal to be detected occurs infrequently over the period in which the autocorrelation is calculated. If this is not the case than the calculated autocorrelation will not resemble the true autocorrelation of the noise, and the technique would fail. It may be necessary to reject an autocorrelation calculated over a period when the signal is present, but in order to evaluate the overall system it is felt that this small aberration may be tolerated.

A more practical analysis of the overall detection system is given in Chapter 8.

8. A PROPOSED DETECTION SYSTEM

8.1.INTRODUCTION

It was shown in Chapters 6 and 7 that it is possible to create a generalised signal detection system which would measure the noise in an input signal and derive the appropriate matched filter. The generation of this filter may be performed once at the start of a measurement, or at intervals throughout the course of the experiment so that it may be applied to the detection of signals in non-stationary noise. For the case where non-stationary noise is present, the system may be considered as a two level scheme, where the foreground activity is the signal detection and the background activity is the updating of the filter coefficients. Furthermore, in Chapter 4 an advanced portable measurement system was proposed based on a dual processor which is suitable for application to problems with both foreground and background activity. In addition, as part of an RSP system, the solution to a measurement problem would be defined at a high level by standard software blocks.

In this chapter the process of creating a signal detection scheme is examined in a "top-down" manner, that is the definition of the problem is dealt with first, including an analysis of the limits to the type of problem which may be tackled, followed by a discussion on the distribution of labour within the dual processor and some brief notes on the requirements of the transducer used to provide a continuous electrical input to the system from the experimental object.

8.2. DEFINING A SIGNAL DETECTION PROBLEM

In a complete RSP system the software required to fulfil a particular function is generated from standard blocks by the high level definer (see Chapter 4). In the case of the proposed signal detector all that would be necessary is to provide the high level definer with a command which would generate the appropriate code, specify the sample rate and supply a suitable model of the signal to be detected. As the background noise is measured by the filter updating section of the generated code it is not essential for the designer to have extensive prior knowledge of the noise, unless it is desirable to obtain an estimate of the performance of the filter, which will be dependent on the relative power of the signal and the noise. This does not mean that it is not necessary to analyse the problem before performing the measurement, as there is a limit to the types of problem which may be solved by applying this type of

adaptive filter. The constraints on the range of problems may be divided into two classes; those due to limitations in the foreground activity (that is, the filtering routines) and those caused by limitations in the background activity (the filter updating program).

8.2.1. Limitations Due To Foreground Activity

The sequential learning identification (SLI) technique used to determine the appropriate matched filter for a given task provides an impulse response model of the filter. Thus the filter will be implemented by performing the operation of convolution in the time domain. For a sampled data system this is defined as

$$y(NT) = x(NT)*h(NT) \qquad 8.1$$

$$= \sum_{m=0}^{N} x(NT-mT)h(mT)$$

$$= \sum_{m=0}^{N} x(mT)h(NT-mT)$$

Equation 8.1. implies that the input is sampled, and output samples are available, at regular intervals. To avoid a breakdown in the system it is therefore necessary to ensure that the sampling interval, T, is greater than the time taken to perform the convolution operation for an input sample. A full analysis of the computation time required for a convolution routine on a F100-L microprocessor is given in Chapter 10. For an impulse response consisting of 64 coefficients the time taken is approximately 44mS, reducing to less than 11mS if a hardware implemented multiply is used. Thus, if a system with software implemented multiplication is used, the minimum sampling period allowed for such a filter is 44 mS, giving a sampling frequency of less than 23Hz. This implies, because of the sampling theorem, that the maximum frequency present in the input signal must not be greater than about 11Hz. With hardware multiplication frequencies of up to 45Hz may be present in the input.

Obviously, the time taken to perform a convolution is dependent on the number of coefficients in the impulse response. As the number of coefficients in the impulse response is equal to the number of sample points defined in the signal to be detected (because of the way the matched filter works) there exists a relationship between the sample rate and the duration of the signal to be detected. The time taken to perform a convolution operation, Tc, on a particular processor may be written as

$$T_c = T_s + KT_p \qquad 8.2$$

where T_s is the time required to set up parameters prior to performing the convolution, T_p is the time required to perform a single multiplication and addition in the format dictated by the convolution algorithm and K is the number of parameters in the impulse response of the filter. In general, T_s will be insignificant in relation to KT_p, especially if K is large. Now, in order to avoid a breakdown the sampling period, T, must not be less than T_c, therefore

$$T \geqslant T_c$$

$$\therefore \quad T \geqslant T_s + KT_p \qquad 8.3$$

T_s and T_p are constants according to the processor and the convolution algorithm, and T will be determined by the frequency content of the input. Arranging 8.3 to find an expression for K yields

$$K \leqslant \frac{T - T_s}{T_p} \qquad 8.4$$

As K is also the number of points defined in the signal to be detected the duration of the signal, T_D, is given by

$$T_D = KT$$

$$\therefore \quad T_D \leqslant \frac{T - T_s}{T_p} T \qquad 8.5$$

For the convolution algorithm implemented on the F100-L T_s = 60μS and T_p = 685μS. Thus if a signal contains frequencies of up to 75Hz, the sampling period should not exceed 6.5mS and the maximum duration of a signal to be detected is

$$T_D \leqslant \left(\frac{6.5 \times 10^{-3} - 60 \times 10^{-6}}{685 \times 10^{-6}} \right) 6.5 \times 10^{-3}$$

$$\leqslant 61 \text{ mS} \qquad 8.6$$

For an algorithm with a hardware implemented multiplication the maximum duration of the signal to be detected is 246mS. If it is felt that the calculated maximum duration of the signal to be detected is not sufficient, and it is not possible to reduce T_p, then the only option is to increase the sampling period, but this may only be done at the expense of the higher frequency components of the signal.

The effects of foreground activity on the range of applications in which the generalised signal detection system may be employed are two-fold. The time taken to perform the necessary calculations for a single input sample affects the maximum sample rate and determines the maximum duration of the signal to be detected. Obviously, these restrictions only apply in real-time systems, and the generalised signal detection scheme could be employed in a non-real-time environment if all else fails.

8.2.2. Limitations Due To Background Activity

The background activity consists of deriving the coefficients of the matched filter from the model of the signal to be detected and the autocorrelation of the noise. Thus the derivation of the coefficients cannot proceed until the autocorrelation has been calculated. As the "goodness" of a calculated approximation of the autocorrelation to the true autocorrelation is dependent on the number of samples over which the average is taken, this could mean there would be an appreciable time lag between two consecutive attempts to derive the matched filter coefficients. Added to this is the time required for each iteration required to produce a suitable estimate of the filter.

The background activity is, therefore, divided into two parts. The first part requires the collection of enough samples and the performing of the calculations necessary to generate an adequate estimate of the autocorrelation of the noise. The second part is the use of the technique of sequential learning identification to obtain a model of the matched filter. When this is accomplished the filter may be used to try and detect the presence of a signal. If the noise is non-stationary a new model of the filter must be generated at intervals throughout the course of the measurement so that the filter is never excessively sub-optimal. However, to prevent a sub-optimal model of the filter being employed at any time, the model used must not have been derived from data which does not have the same statistical properties as the data which is applied as input to the filter. This implies that the duration of the filter generation activity must not exceed the length of time over which the noise may be considered stationary. As an example, consider an input which is sampled at regular intervals, say, T. For a good approximation to the autocorrelation N samples are required, and the time required to perform the calculation of the autocorrelation is T_A. The identification routine may now proceed, and this takes, say, T_{ID} seconds, after which a model of the filter is obtained, and this is used for a time T_M. Thus the time, T_{STAT}, over which the signal may be considered stationary is governed by

$$T_{STAT} \geqslant NT + T_A + T_{ID} + T_M \qquad 8.7$$

This is shown diagrammatically in figure 8.1. Although the time over which the filter is actually used, T_M, is part of the foreground activity it is, in fact, related to the processing time needed for the background activity. The time required to generate a new estimate of the filter parameters, T_G, is given by

$$T_G = NT + T_A + T_{ID} \qquad 8.8$$

Until the new estimate of the filter is complete the old estimate must be employed as the actual filter, and this implies that

$$T_M \geqslant T_G \qquad 8.9$$

FIGURE 8.1 RELATIONSHIP BETWEEN THE TIME REQUIRED FOR THE BACKGROUND ACTIVITY AND THE RATE OF CHANGE OF THE NON-STATIONARITY IN THE NOISE.

Referring back to the relationship given in 8.6 we may now write

$$T_{STAT} \geqslant 2T_G \qquad 8.10$$

Thus the maximum rate at which the non-stationarity may alter the quality of the noise is determined solely by the time taken to perform the background activity.

The above approach has tackled the problem of the maximum rate of change possible in the noise in a very straightforward manner. The situation may be greatly improved by bearing in mind the fact that the processes of filter generation and the employment of the filter will, in a dual processor system, be running concurrently. Therefore, provided it is possible to store all the input samples arriving during the time taken to generate the new filter , it is possible to determine the new coefficients of the filter and, when this is complete, run the previously collected samples through the new filter. While these old samples are being processed new samples are taken in, stored and used to generate the next estimate of the filter. As the filter is only used on samples taken in during the period T_G this effectively makes $T_M = 0$. Therefore, from relationship 8.7

$$T_{STAT} \geqslant NT + T_A + T_{ID} = T_G \qquad 8.11$$

Thus the maximum rate of change of the non-stationarity present has been doubled, provided enough store is available to hold the input samples taken in over the period T_G.

The major disadvantage of such a technique is the extremely long delay between receiving an input sample and generating the related output sample. This delay will, of course, be equal to T_G.

Further improvements to the maximum rate of change in the noise may be gained by reducing T_G. The only practicable way of achieving this is to reduce T_A, which is the time required to produce an estimate of the autocorrelation after a suitable number of samples have been obtained. This reduction may be obtained by calculating the "partial sums of products" every time a new sample is input, thereby minimising the amount of processing needed after the last sample required is input. This technique is examined more thoroughly in Chapter 9, but briefly what

happens is this: when a new sample is input it is multiplied by itself and the product is added into the current total for the autocorrelation point R(0). It is then multiplied by the previous sample, and this product is added into the current total for R(1). This process of multiplying the latest sample by previous samples and adding the product into the appropriate partial sum in the autocorrelation continues until all the points in the autocorrelation have been updated. When this is complete, a new sample may be input. When enough samples have been input for the estimate of the autocorrelation to be acceptable, each point in the autocorrelation is divided by the number of samples to form the average value. The time taken to perform this division (which may be implemented as a binary shift if the number of samples is selected to be a power of two) is the new T_A.

To avoid a build up of samples during the performance of the intermediate calculation which form the estimate of the autocorrelation, the time taken to form the partial sums of products must not exceed the sampling period. Thus it might appear that this method of calulating the autocorrelation would also effect the sampling rate. However, as the number of points required in the autocorrelation is the same as the number of parameters in the filter, and both convolution and correlation require the formation of sums of products, the time taken to perform the intermediate sums for the autocorrelation will not be greater than that required to perform the convolution operation for each input sample. It follows that the sample rate will still be determined by the foreground activity.

8.2.3. The User Interfaces

An instrument designer who wishes to make use of the facilities of the proposed RSP system gains access to the software units he requires through the high level definer (HLD). The HLD is a program which, when given a set of commands, generates the program to be run on a "black box" instrument, which is in this case a dual-processor based RSP. The HLD may take the form either of a compiler, whereby the instrument designer writes a program consisting mainly of calls to standard software units, or of an interactive conversational language, where the final program is generated after a dialogue between the designer and the HLD. In either case, the restrictions on performance given above may be calculated by the HLD and warnings issued if an attempt is made to define a non-implementable instrument.

The first task facing the designer is to inform the HLD of the software block or blocks that he requires, in this case the two blocks forming the foreground and the background activity of the generalised signal detector. These blocks will often consist of smaller units, for example units performing the functions of convolution and correlation, but this will be transparent to the user. Given the function to be performed, the HLD is then in a position to decide if a given task if feasible when the necessary parameters are given.

The first parameter required is the sampling period. There are many reasons why the sampling period should be presented first, not the least of which is the fact that it is, above all, the one parameter which the designer must get absolutely correct, as neither the HLD or the RSP when it is performing the measurement can indicate that a nonsensical sample rate has been requested. The sample rate, in the limit, is a function of the maximum frequency present in the input signal before the ADC takes place, and neither the HLD or the RSP has knowledge of the signal at that point.

Once the sampling period has been established the HLD is in a position, provided it has been previously supplied with the appropriate computation times, to indicate the maximum duration of the signal to be detected according to the relationship given in 8.5. If this time proves insufficient, the instrument designer has no option other than to redefine the problem. This may be achieved by increasing the sample period if possible, or by detecting only a portion of the signal. When a combination of sample period and the duration of the signal is obtained which is acceptable to both the designer and the HLD, it is then possible to move on to the next stage in the process, which is the definition of the signal which is to be detected.

The model of the signal to be detected is presented to the HLD as a series of samples taken from the signal at intervals equal to the sample period, in much the same way as the noise-free signal would appear to the RSP after it had been sampled and quantized. However, as it is the waveshape which is important, and not its amplitude, the model of the signal may be amplified to make full use of the computer wordlength. This assists the speedy generation of the filter parameters whilst assuring that the coefficients are not restricted to only two or three meaningful bits of the computer word. It is also necessary to provide the HLD with some

indication as to the number of samples in the model of the signal, and when this is available the HLD may then calculate the minimum time over which the input signal may be considered stationary, according to either relationship 8.10 or 8.11, depending on whether it is considered reasonable to store all incoming samples while the new estimate of the filter is being derived. If it is known that the input signal may be considered stationary over a period greater than that calculated by the HLD then there is a good chance that the filter will perform according to expectations. In other cases, where it is known that the signal will be non-stationary over the period calculated by the HLD or where the nature of any non-stationarity is unknown, a trial and error tactic would have to be employed, first by using the generated filter and, if the performance of the filter is inadequate, thereafter re-defining the problem by, for example, detecting only a portion of the signal.

Once the sample rate and the signal to be detected are defined, and all restrictions have been met, the HLD may then compile the necessary software units to form the program which will be run on the RSP. In general there will be more to the problem than this, in particular the definition of the course of action to be taken once a signal has been identified, but this will vary considerably between experiments although, again, there will be a number of standard software units available to aid the designer.

8.3. THE DIVISION OF LABOUR WITH THE DUAL PROCESSOR

After the HLD has received an acceptable series of commands defining a particular function it then produces a program which is to run on the RSP. With most compilers this is a very straightforward task, as the target machine is usually a single processor where instructions are executed sequentially. However, the target machine for the HLD is a dual processor configuration where instructions may be executed concurrently. Therefore there must be some decision making mechanism for allocating tasks to the two processors. The simplest way of achieving this, from the point of view of the person who designs the HLD, is that the user must specify which of the two processors will perform a given task, or specify two programs (one for each of the processors). While this gives the user greater flexibility in designing an instrument it means that the RSP may no longer be considered as a "black-box" instrument, and the user must have some knowledge of the internal structure and

the problems associated with shared resources within multi-processor systems. An alternative would be to make the software blocks as main processor or I/O processor activities, with an option to override these default allocations. This would provide the non-expert with a method of easily defining an instrument while allowing the expert user the facility to optimise the implementation of a particular task. The ability to override the HLD default allocations is desirable bacause, in the first place, a large number of applications would divide tasks unequally between the two processors such that one processor would be idle for large periods, and secondly because a background activity would often require the use of software blocks which, in themselves, would be considered as main processor activities. This is true of the generalised signal detector, where the sequential learning identification program requires the convolution function. Furthermore, there are some activities, for example correlation, which may be considered as either main or I/O processes.

In the case of the generalised signal detector the division between the main process and the background activity is very clear. The main processor is performing a convulution operation in real-time and the I/O processor is continually updating the estimate of the filter. However, the background activity of generating a new estimate of the filter takes much longer to perform than the foreground activity of convolution. At the same time, the technique of sequential learning identification requires the operation of convolving the autocorrelation of the noise with the current estimate of the filter to give an indication of the error between the true and the estimated values of the filter coefficients (see Chapter 7). In order to balance the processing demands between the two processors in the RSP it might be suggested that the main processor could perform the convolution operation for the I/O processor, leaving the latter free to calculate other parts of the algorithm. While this is a possibility, there are several reasons why this is not desirable. First, the main processor is working in real-time and any further processing load would degrade the system's performance by reducing the maximum sample rate (in this case by more than a factor of 2). The second point concerns the degradation of performance of a dual processor system when there is contention over a shared resource, in this case the common memory block. As the amount of communication between the two processors increases, so does the possibility of contention for access to the common memory, which leads to a worsening of performance. As the background activity affects only the maximum

rate at which the statistics of the incoming signal may alter, while the foreground activity directly influences the maximum sample rate, it is better to keep both activities separate as far as possible.

The arrangement of the dual processor is such that the main processor has no access to any of the I/O peripherals. Therefore all I/O and any pre- or post-processing requirements must be fulfilled by the I/O processor. The steps taken in the generalised signal detector, in each processor, are shown in figure 8.2. The I/O processor takes in a sample and loads it into a cyclic buffer, making the fetching of samples for the convolution operation more simple. With this done, the main processor performs the convolution function and produces a result, which will be analysed by a comparatively simple routine, called here "DECIDE". This routine will indicate the presence or absence of a signal to the I/O processor, which will act accordingly. While this is proceeding the I/O processor will be performing one of two tasks. First, an estimate of the autocorrelation will be calculated, and when this is complete the sequential learning identification algorithm will be entered. Note that in figure 8.2 the problem is tackled in the straightforward manner proposed in the beginning of section 8.2.2., but if the improved technique suggested later on in that section is required, all that is necessary is to incorporate a more complex buffering scheme when the sample is first input.

For most of the time the I/O processor will be performing the sequential learning identification procedure which will not, in general, be related to the sample rate. At the same time the I/O processor must input samples at regular intervals. Thus it is imperative that the I/O routines for the ADC be interrupt driven, for there is no mechanism by which the I/O processor can decide when to break off from running the identification algorithm to poll the ADC. This is an important point, as although it is nearly always desirable to perform I/O by the use of interrupts, in this case it is absolutely essential to the running of the program.

8.4. THE TRANSDUCER REQUIREMENTS

Although not strictly part of an RSP system, the design and specification of the transducer which provides the input to the RSP affects the performance of the overall system and, if insufficient care is taken in the transducer design,

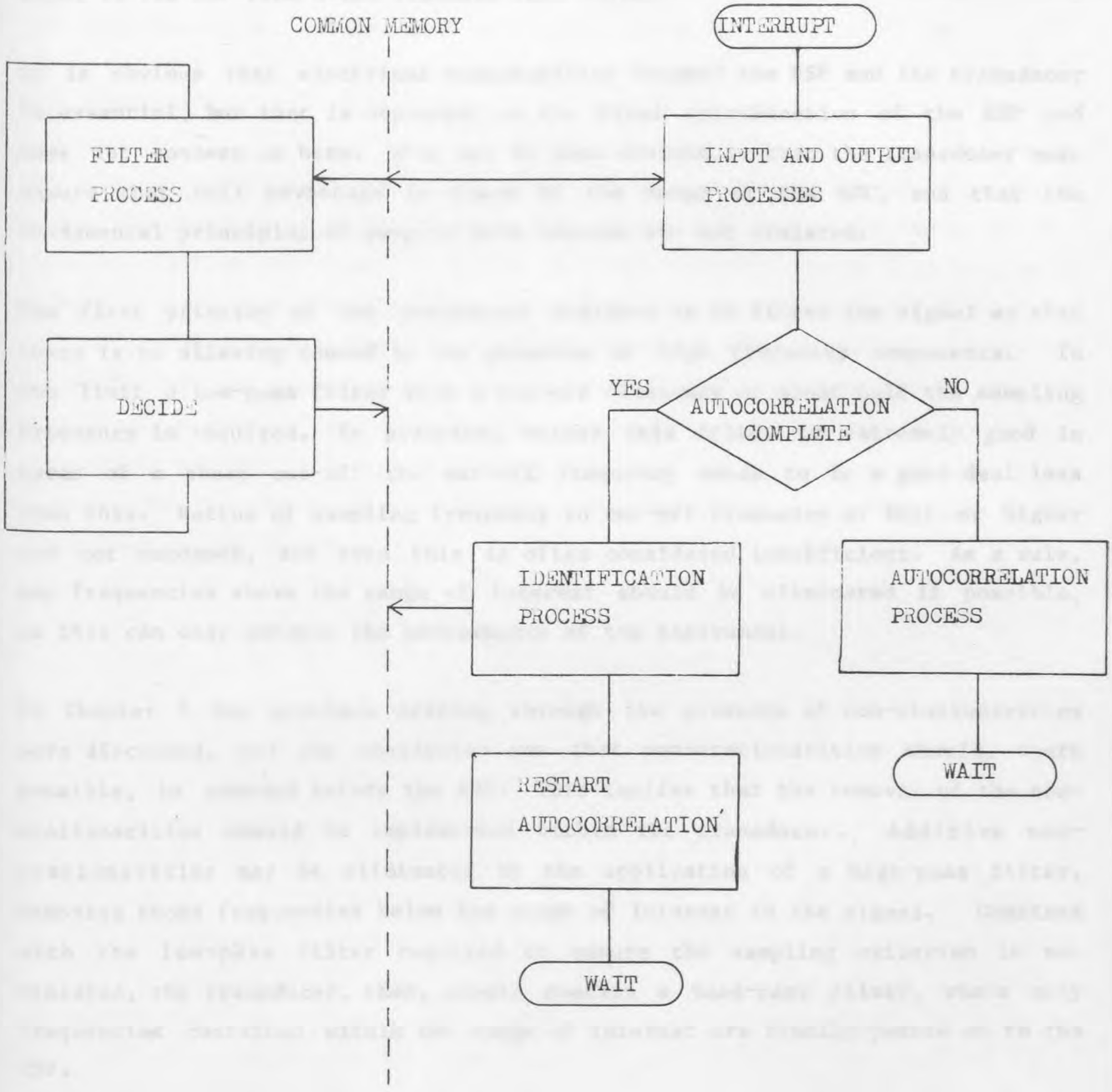


FIGURE 8.2 THE DIVISION OF LABOUR WITHIN THE DUAL PROCESSOR FOR THE GENERALISED SIGNAL DETECTOR

may possibly produce nonsensical results. Basically, the RSP is a generalised system and therefore the transducer must be tailored to produce a standard input to the RSP from a non-standard test object.

It is obvious that electrical compatability between the RSP and the transducer is essential, but that is dependent on the final specification of the RSP and does not concern us here. What may be less obvious is that the transducer must ensure that full advantage is taken of the range of the ADC, and that the fundamental principles of sampled data systems are not violated.

The first priority of the transducer designer is to filter the signal so that there is no aliasing caused by the presence of high frequency components. In the limit a low-pass filter with a cut-off frequency of about half the sampling frequency is required. In practise, unless this filter is extremely good in terms of a sharp cut-off the cut-off frequency needs to be a good deal less than this. Ratios of sampling frequency to cut-off frequency of 10:1 or higher are not uncommon, and even this is often considered insufficient. As a rule, any frequencies above the range of interest should be eliminated if possible, as this can only enhance the performance of the instrument.

In Chapter 5 the problems arising through the presence of non-stationarities were discussed, and the conclusion was that non-stationarities should, where possible, be removed before the ADC. This implies that the removal of the non-stationarities should be implemented within the transducer. Additive non-stationarities may be eliminated by the application of a high-pass filter, removing those frequencies below the range of interest in the signal. Combined with the low-pass filter required to ensure the sampling criterion is not violated, the transducer, then, should contain a band-pass filter, where only frequencies contained within the range of interest are finally passed on to the RSP.

Multiplicative non-stationarities are, as pointed out in Chapter 5, a more difficult problem to solve but again they should be removed before the ADC takes place. One solution is to use non-linear amplification (for example logarithmic), while bearing in mind the effect this has on the signal and adjusting the definition of the problem accordingly.

It should be remembered that any erroneous data or loss of accuracy occuring before the ADC will cause a propagation of errors throughout the defined instrument which are neither recoverable nor (in general) detectable by the

instrument. Data will be considered valid even if there is aliasing or the signal is represented by only two or three bits. It is essential that great care is taken when designing the transducer if the work put in to specifying and constructing an instrument is not to be wasted.

8.3. CONCLUSIONS

The proposed detection system provided a quick and simple method of producing an instrument which will detect a given signal. The instrument designer only needs to specify the sample rate and the signal which is to be detected, while liasing with the transducer designer to ensure that the data presented is valid and as accurate as possible. Once the problem is defined the HLD generates the necessary code and allocates the tasks to each of the two processors in the RSP according to whether the tasks are I/O, real-time or non-real-time. As it is a generalised system, it is unlikely to provide the ideal solution to any given problem, but is may produce a working solution and a large reduction in the time taken for development.

9. IMPLEMENTATION AND PERFORMANCE

9.1. THE OBJECTIVES OF THE SYSTEM

The generalised signal detector as proposed above consists of a foreground, real-time process and a background, non-real-time process. For efficient implementation of this measurement technique a system consisting of (at least) two processors is required, such as the advanced portable measurement system proposed in outline form in Part 1 of this thesis. However, the practicalities of multi-processor configurations are such that a thorough examination of the loss of efficienty caused by communication overheads and resource contention is required. Thus the development and analysis of a dual processor system was itself the subject of research contemporaneous to that contained in this thesis. Such research into the problems of multi-processor configurations is essential if one is to provide a solid foundation on which to develop the advanced portable measurement system, and the results of this research may be found in reference 34. This need not concern us here, however, as the existence of an efficient dual processor has been assumed throughout the development of the generalised signal detector, although it must be remembered that the efficiency of the dual processor system will affect the final performance of the signal detector. Indeed, this is one of the reasons why the signal detector has been developed on a single processor system. The generalised signal detector is a novel measurement method, and as such it is desirable to examine both its efficiency and applicability to implementation on a microprocessor system. Should this analysis be performed on a dual processor these characteristics would be inextricably involved with the inefficiencies of the system, and the performance of the signal detector would then be dependent upon the structure of the dual processor. It was considered a better strategy to obtain some form of absolute performance measurement (albeit absolute on only one type of processor) and then allow for degradation when the program is implemented on a dual processor system.

The desire for a degree of absoluteness in obtaining a measurement of performance is not the only reason for implementing the prototype signal detector on a single processor system; there are good practical reasons for doing so. The signal detector program, although not excessively large in terms of the storage area required, is quite complicated and contains a number of nested program loops. It would be unnecessarily long-winded to attempt to develop such a program without the aid of a suitable debug package including facilities such as breakpoints and the listing of the contents of memory

locations. Furthermore, the F100-L microprocessor (on which the program is to run) has no multiply or divide instruction, and the user must either write his own routines for these operations or use the routines provided in the F100-L mathematical library. Finally, the advanced portable measurement system does not have extensive backing storage facilities (floppy disk, for example) so that the retention of results for further analysis is difficult. For these reasons it was decided that the host system for the development of the generalised signal detector should be the F100-L DOS development system. It is felt that the major objectives of the prototype - development and performance testing - would be better served if it were implemented on the development system.

9.2. THE F100-L DEVELOPMENT SYSTEM

The F100-L development system consists of a single F100-L 16-bit microprocessor, 32K words of RAM, a V.D.U./teletype/lineprinter interface and dual floppy disk drive and interface. The system software includes a software control module (SCM) which performs all system I/O functions, a disk-based operating system (DOS), assembler, link-editor, text editor, debug package, mathematical and I/O subroutine libraries and a library manager. An optional CORAL 66 compiler is also available. The SCM is usually accessed as part of DOS, but it may be used separately. All I/O to system peripherals is streamed; that is, system or user programs make transfers on a stream number, and this stream is allocated to a peripheral by the user before a transfer takes place. User peripherals, for example ADC/DAC, magnetic bubble memory interface, etc., cannot be streamed but the provision of a library manager allows driving routines to be included in a standard library.

A further facility on the F100-L development system is the control handset, which is a miniaturised front panel. This enables memory locations and the program counter to be examined and altered, the monitoring of the data/address bus, run/stop/reset of the processor and "single-shot", i.e. execute only one instruction at a time. However, as these facilities are available in the debug package (with the exception of processor run/stop/reset, which should not be used when control is in DOS) it is advisable to use the control handset sparingly, at it gives no record of any alterations made to locations, making mistakes difficult to trace.

It was stated above that the objectives of the prototype are two-fold. The first objective is program development, and this is itself in two parts; program writing and alteration and program debugging. Unfortunately for the purposes of writing and altering programs the relevant system software has been written in a rather old-fashioned way, making these stages of development somewhat more time-consuming than would otherwise be the case. To write a program to disk the user must first create a file and specify its length in allocation units (1A.U. = 256 characters). Until the user develops the "knack" of estimating the size of the program this frequently means an underestimation of program size (and consequently running out of file space before the program is fully input) or an overestimation leading to a loss of valuable disk space. The program is written to disk by "copying" characters from the v.d.u./teletype (or other input device) to the appropriate file. Program alteration, by the use of the text editor, is similarly old-fashioned. Again, a new file must be created and its length specified (unless the edited program is to be written back to its original file - a risky practice and one which is not recommended). The program is then read into store where it may be edited. Should the program be too large to fit into the store allocated to the text editor a number of "read" commands must be issued, each segment in store being "kept" before a new segment is read. Once the program is in an acceptable form it may be assembled and link-edited to give an object listing suitable for loading into store.

The program may be tested by loading both the program and the debug package into store and entering the debug program at a suitable point (depending on which streams are used for debug I/O). The user may now examine and alter locations (including the accumulator and program counter), set "traps" (breakpoints) and run or single-step through the user program. The trap facility is extremely useful, as the trigger may be a location (whether accessed or executed) or an instruction. Multiple traps are allowed. It is advisable to use a teletype or printer with the debug package rather than just a v.d.u. as a hard copy is invaluable for checking on previous alterations made to the program when it is in store which may be mistakes or "patches" (small program units outside the original program area which are jumped to at appropriate points in the program. It is undesirable to let the number of patches become excessive, but it is not feasible to edit the program every time an error is found).

The second objective of the prototype is the collection of results. Many of the results - autocorrelations, impulse responses and filter input and output - are more meaningful if presented in graphical form. Unfortunately, the F100-L development system does not have graphics capability, although at the time of writing work aimed at providing a limited graphics capability was at an advanced stage. The types of result quoted above may be represented as an amplitude versus time plot, and this was how the results were finally obtained. When a result was obtained that required plotting it was written as a series of 8-bit bytes to a disk file from which it may be loaded into store and finally output through a DAC to a y-t plotter at a later date. This is achieved very simply by using the system I/O subroutines. Other results - timing information, for example - do not directly require any of the facilities available on the development system, except for program storage and loading.

9.3. THE PROTOTYPE STRUCTURE

The generalised signal detector consists of two separate entities known as the foreground and background activities. The background activity itself consists of two parts, the first of which is the calculation of the autocorrelation of the noise and this is followed by the sequential learning identification (SLI) routine. Furthermore, it is not known in advance how many iterations are required by the SLI technique to obtain a reasonable estimate of the filter coefficients. In additions, it was felt desirable to observe the response of the SLI technique when "ideal" white noise is assumed. Obviously, the autocorrelation of ideal white noise cannot be calculated, thus a preset value is given to the autocorrelation so that the SLI may take place without an autocorrelation being calculated previously. Finally, the SLI requires an autocorrelation (either preset or calculated) and the filter routine requires an impulse response as generated by the sequential learning identification. It follows that each of the three parts should be contained within one program. For these reasons each of the three parts is contained as a sub-program with a master routine running one of them according to an input command. The total program length, including data areas, is less than 2K words.

9.3.1. The Master Segment

The major task of the master segment is to read a single letter command on stream 1 and initiate a jump to the relevant sub-program. In addition it sets up the workspace area for the system subroutines and initialises the I/O channels. It also sets up the autocorrelation data area to a series

of preset values corresponding to the autocorrelation of "ideal" white noise, and creates a table of squares. It achieves the latter in a simple manner by the use of the formula

$$(I + 1)^2 = I^2 + 2I + 1 \qquad 9.1$$

Initially I = 0 and therefore $I^2 = 0$ such that the first iterations generate the sequence

$$(0 + 1)^2 = 1^2 = 0 + 0 + 1 = 1$$
$$(1 + 1)^2 = 2^2 = 1 + 2 + 1 = 4$$
$$(2 + 1)^2 = 3^2 = 4 + 4 + 1 = 9$$

and so on. The 256 numbers generated are placed in consequtive locations starting at the address labelled "SQTAB", so that the square of an 8-bit number may be found by

$$N^2 = \text{contents of (SQTAB + N)} \qquad 9.2$$

This greatly improves the performance of the SLI routine which requires the sum of the squares of the filter input vector.

The master segment is, then, a relatively small part of the program (it consists of 76 words of memory) which initialises data areas and then calls the appropriate sub-program according to a single letter command. The commands and the associated routines, are presented below.

9.3.2. The Command "A" : Perform Autocorrelation

The autocorrelation subroutine takes a stream of "samples" in on stream 3, forms the double-length autocorrelation in the data area RXX and stores the most significant 16 bits in a plotter area, from which it may be loaded to disk (see below). For an N point autocorrelation a total of N-1 samples are read in before any calculations take place, so that each point in the autocorrelation is the sum of the same number of products; in this case 8192 samples are used, but it is possible to obtain a reasonable estimate of the autocorrelation from a smaller number of samples.

The autocorrelation is calculated as follows. If the latest sample input is, say, x_n then each point in the autocorrelation is updated by

$$R(i)_n = R(i)_{n-1} + x_n x_{n-i} \qquad 0 < i < n \qquad 9.3$$

until enough samples are input. Normally, in order to obtain a correct value for the autocorrelation, each point in the autocorrelation would then be divided by the number of samples but for this application only the shape of the autocorrelation is important. Thus R(0) (which is the largest of the points) is normalised and the other points shifted by the same amount. This method of calculating the autocorrelation only requires N samples to be stored, this being achieved by using a cyclic buffer.

In order to improve the performance of the autocorrelation routine it was noted that the input consists of 8-bit unsigned numbers. A special routine was written which multiplies two 8-bit unsigned numbers to give a 16-bit positive result rather than use the longer system subroutines which implement a 16 x 16 bit multiplication for signed numbers.

The autocorrelation subprogram forms a major part of the bulk of the prototype consisting of 147 words of memory. It also takes the most time, as an autocorrelation is not complete until 8255 (8192 + 63) samples have been read in. This is obviously an undesirable state of affairs, and an alternative strategy is discussed in Chapter 11.

9.3.3. The Command "E" : End Program Run

This is a small routine (6 words) which closes all I/O streams and returns control to DOS via a controlled hold; it should be noted that all programs run under DOS must terminate with a subroutine call which causes a hold rather than execute a HLT instruction, which causes the processor to stop.

9.3.4. The Command "F" : Implement Non-recursive Filter

The filter subprogram is implemented in two parts. One part is a sum of products subroutine, which is called by another part which handles the I/O. Input is taken in on stream 5 and the output appears on stream 4, so that the filter does not operate in real-time, although an early version of the program did use the ADC/DAC for I/O and was used to obtain timing information (see later). The reason for the change to streamed I/O was

purely for the sake of simple storage of results and the precise repeatability of the input signal. The number of coefficients in the filter and the address-1 of these coefficients are set up in a parameter data block, and throughout the examination of the prototype the number of coefficients has been assumed to be 64. There is no particular reason for this except that, at least in computing terms, it is a nice round number and it is a sufficient number of points to represent adequately the spike-and-wave complex in the EEG.

The first task of the sub-program is to clear the double length result locations and open the I/O streams. A single sample is input and written to each location in a cyclic buffer, in order to set up a "steady-state" value. The reason for doing this is to prevent the plotter pen from swinging wildly when the result is finally plotted; it does not affect any results after the 64^{th} sample (due to the limited memory of the filter) while results previous to the 64^{th} sample must be considered as unreliable anyway. Samples are then input and output until an EOF (end of file) character is read on stream 5, whereupon the I/O streams are closed and control returned to the master segment. When a sample is read it is loaded into a cyclic buffer and the convolution routine is called. This fetches samples from the cyclic buffer (latest sample first) and multiplies them by h(0), h(1), ..., h(63) in order. The double length sum of the 64 products is returned to the calling routine, which adjusts them by a shift so that only the most significant 8 bits remain. The length of this shift depends on the number of filter coefficients, say N, and the number of bits required to hold the impulse response. As the impulse response is provided by the SLI routine it may not be known how many bits are required to hold the impulse response, although as the SLI technique converges to the true coefficients of the filter then, provided the error correction coefficient is less than unity, the number of bits required should not exceed 8. However, assuming the number of bits required for the impulse response is, say, K then the shift displacement is given by

$$\text{bits} = 16-k-\log_2 N \qquad 9.4$$

where a negative number implies a right shift and a positive number a left shift. For 64 coefficients and an 8-bit impulse response this means a left shift of 2 places is required, and this has been found to be satisfactory.

The filter implementation routine is comparatively small, with 69 words being required for the calling subroutine and 34 words for the convolution.

9.3.5. The Command "O" : Find The Optimal Filter

The SLI routine uses the autocorrelation in the data area RXX (which may be either a set of preset values or a calculated autocorrelation) and the model of the signal to be detected in the data area SIG to update the present estimate of the filter coefficients. To achieve this it must implement the equation

$$\hat{g}_n = \hat{g}_{n-1} + (y_{n-1} - \hat{y}_{n-1})x_{n-1} / x_{n-1}^T x_{n-1} \qquad 9.5$$

where $\hat{g}$ is in the data area IR (impulse response), y is the signal to be detected, $\hat{y}$ is the output of the present estimate of the filter and x is the double-sided autocorrelation. Although only one side of the autocorrelation is calculated (i.e. those points for which τ is positive) it is an even function so that

$$R_{xx}(-\tau) = Rxx(\tau) \qquad 9.6$$

and therefore both sides of the autocorrelation are known.

Initially the points $R_{xx}(-63)$ to $R_{xx}(-1)$ are loaded into the cyclic buffer from which they are used as input for the current estimate of the matched filter. A pointer is set to the start of the data area SIG so that the signal to be detected may be accessed, and another points to the start of RXX so that $R_{xx}(0)$ to $R_{xx}(63)$ may be fetched when required. Having accomplished this the program enters an external loop which performs a single iteration 64 times so that each value in the signal to be detected is used to provide the new estimate of the filter.

Throughout the calculation all operands are scaled to ensure the minimum loss of information through underflow. Scaling also allows data to be shifted so that the division operation can never overflow. The results are descaled before they are added to the estimate of the impulse response. Intermediate results are built up in an 8 word workspace data

area, and operands are placed in this workspace in such a way as to minimise their movement throughout the data area as dictated by the parameter transfer rules of the multiplication and division subroutines.

The algorithm proceeds in this manner. The next point in the autocorrelation is loaded into the cyclic buffer, which is then convolved with the present estimate of the filter to produce an estimated output, $\hat{y}$. This is subtracted from the appropriate point in the signal to be detected and the result is multiplied by the error correcting coefficient, α. The sum of squares of the input vector, $x^T x$, is then calculated by using a look-up table. Finally, the product of α and $(y-\hat{y})$ is divided by $x^T x$ and the program enters an inner loop.

The algorithm's inner loop multiplies the result of $\alpha(y-\hat{y})/x^T x$ by each point in the resulting error correction vector to the impulse response. Control then passes to the external loop until 64 iterations have been completed. Finally, the new estimate of the filter is loaded to the plotter area, from where it may be saved on a disk file, and control returned to the master program.

The SLI routine requires 155 words of store and also uses the convolution (34 words), multiplication (70 words) and division (104 words) subroutines.

9.3.6. The Command "P" : Plot Results

The autocorrelation and filter identification sub-programs place a copy of their results in a data area from which the user may store them on a disk file by typing the "P" command. This initiates a dummy plotter routine, which takes the 16-bit data in the plotter data area and manipulates it to take the maximum advantage of the 8-bit wide DAC through which the results will be displayed at a later date.

The manipulation of the data begins with a search to find the largest and smallest values in the data area. The smallest value is subtracted from all points in the data area and the location holding the largest value. The location which contains the largest value is then shifted until the top bit of the 16 bit word is set and a count is kept of the number of shifts required to do this. Finally, all the points in the data area are shifted by the same amount as the count and then shifted right 8 places so

that they occupy the least significant 8 bits of the word. The data area may now be written to a disk file as 8 bit binary data (not characters).

The results may be written to the DAC, and thence to a y-t plotter, by the use of the program DC. This program is loaded into store and location 200 is set to -(the number of points) by the handset. Stream 1 is for commands and stream 3 is the disk file containing the results. Running the program from 2048 loads the results into store from where they will be written to the DAC every time a "p" is input on the command stream. Should any other character be input on stream 1 then the program will return control to DOS.

The plotter subprogram requires 93 words of store.

9.4. TIMING INFORMATION

For the prototype generalised signal detector the filter and autocorrelation sub-programs were written so that input and output are stored on disk files. Such an implementation does not allow the measurement of timing information which is relevant to the operation of these program units in a real-time system as there will be extensive and variable delays inclurred when the files are accessed. In order to obtain valid timing information, therefore, it was necessary to use an early prototype program which differs from that described above only in the respect that input and output is performed through the ADC/DAC.

There are two signals on the ADC/DAC which enable timing information, in the form of the maximum allowable sample rate, to be obtained. The first is an external clock derived from a pulse generator which dictates the rate at which an analogue input is sampled and digitised. The second signal is a "sample-read" which gives a short pulse each time the processor requests a new sample. Thus if these signals are presented to the two channels of a fast (200MHz) two-channel oscilloscope one may see if there is one read pulse for every clock cycle. This is achieved by synchronising the oscilloscope on the clock signal and, if there is one read for each clock pulse, the two signals will be synchronised and the traces will appear to be stationary. If, however, there is not one read for each clock pulse (i.e. the sample rate is too fast) then the sample read signal will appear to "slip" across the oscilloscope screen.

The filter subprogram consists of a sample read section and a convolution subroutine. The time required by the sample read section will be constant, but the convolution subroutine will require a time proportional to the number of filter coefficients. The total time required for the filter subprogram may be written as

$$T_c = T_s + KT_p \qquad 9.7$$

where T_c is the total computation time, T_s is the sample read and set up time, T_p is the time taken for each coefficient and K is the number of coefficients. T_s may be easily calculated from the manufacturer's timing information as it consists of only a few instructions. It transpires that T_s is approximately 137μS. By the measurement technique given above it was found that, for a filter with 64 coefficients, T_c was 44mS. Thus we have

$$44 \times 10^{-3} = 137 \times 10^{-6} + 64\ T_p \qquad 9.8$$

By simple algebraic manipulation we can see that T_p = 685μS. Thus, for a filter of K coefficients implemented on the F100-L the minimum sample period is given by

$$T(\text{in } \mu S) = 137 + K(685) \qquad 9.9$$

A similar analysis can be made with the autocorrelation sub-program. In this case the sample read section requires 106.5μS, and the measured minimum sample period for a 64 point autocorrelation is 36mS. This gives a computation time of 545uS for each point in the autocorrelation, that is

$$T(\text{in } \mu S) = 106.5 + K(545) \qquad 9.10$$

where T is the total computation time for a K point autocorrelation.

The above results are shown graphically in figure 9.1 for K less than 10. It can be seen that the filter subprogram is always the slower of the two processes, and therefore it will be this which determines the maximum sampling

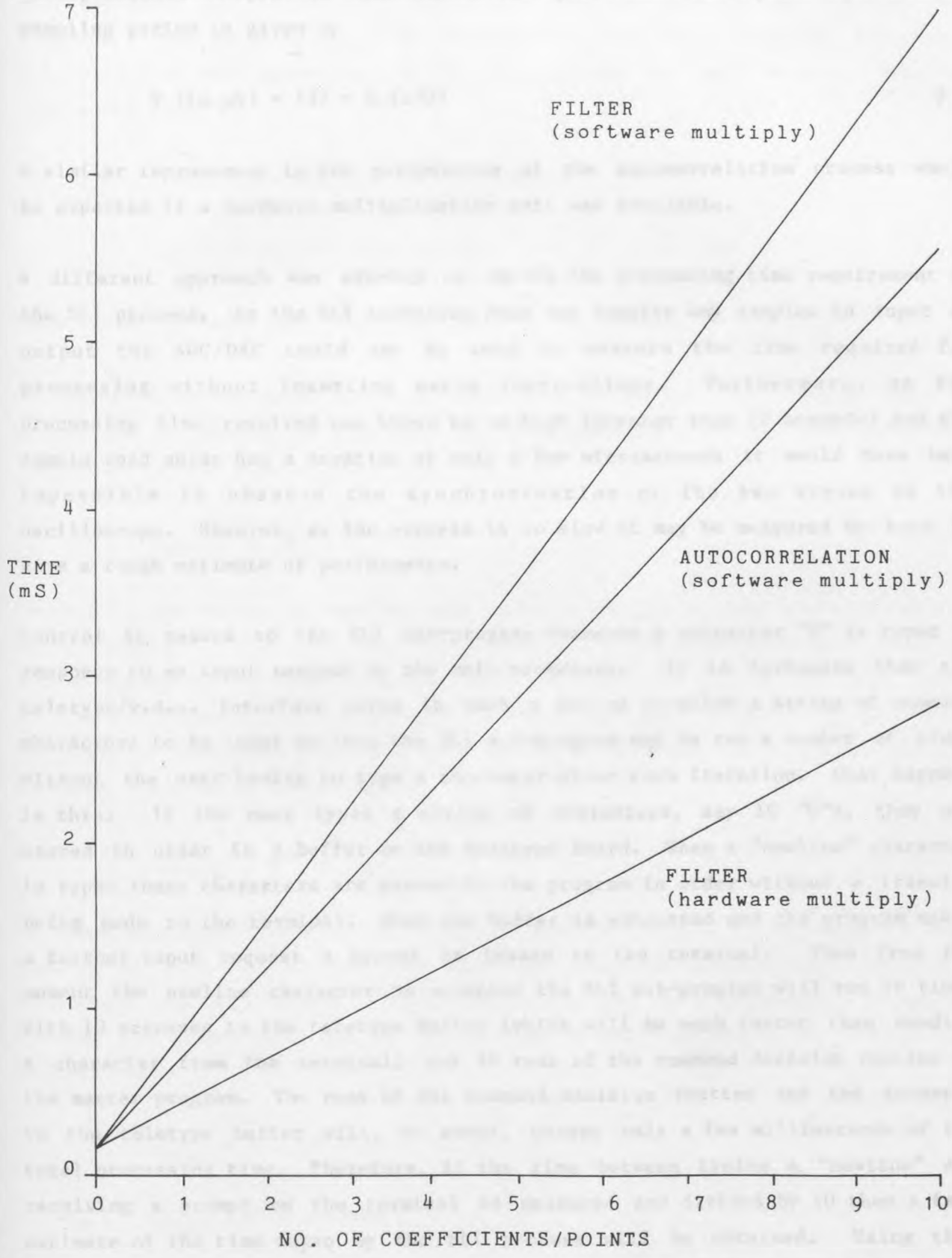


FIGURE 9.1 NUMBER OF COEFFICIENTS OR POINTS VS. MINIMUM SAMPLING PERIOD.

rate. Figure 9.1 also shows the relative improvement if a hardware multiplication is provided for the filter process. In this case the minimum sampling period is given by

$$T \text{ (in } \mu S) = 137 + K\ (170) \qquad 9.11$$

A similar improvement in the performance of the autocorrelation process would be expected if a hardware multiplication unit was available.

A different approach was adopted to obtain the processing time requirement of the SLI process. As the SLI technique does not require any samples to input or output the ADC/DAC could not be used to measure the time required for processing without inserting extra instructions. Furthermore, as the processing time required was known to be high (greater than 10 seconds) and the sample read pulse has a duration of only a few microseconds it would have been impossible to observe the synchronisation of the two traces on the oscilloscope. However, as the process is so slow it may be measured by hand to give a rough estimate of performance.

Control is passed to the SLI sub-program whenever a character "0" is typed in response to an input request by the main processor. It is fortunate that the teletype/v.d.u. interface works in such a way as to allow a string of command characters to be input so that the SLI sub-program may be run a number of times without the user having to type a character after each iteration. What happens is this. If the user types a string of characters, say 10 "0"s, they are stored in order in a buffer on the teletype board. When a "newline" character is typed these characters are passed to the program in order without a transfer being made to the terminal. When the buffer is exhausted and the program makes a further input request a prompt is issued to the terminal. Thus from the moment the newline character is accepted the SLI sub-program will run 10 times with 10 accesses to the teletype buffer (which will be much faster than reading a character from the terminal) and 10 runs of the command decision routine of the master program. The runs of the command decision routine and the accesses to the teletype buffer will, at worst, occupy only a few milliseconds of the total processing time. Therefore, if the time between typing a "newline" and receiving a prompt on the terminal is measured and divided by 10 then a fair estimate of the time taken by the SLI process will be obtained. Using this method a figure of just under 12 seconds was obtained for 1 run of the SLI process.

Once again it is possible to estimate the improvement in performance when a hardware multiply/divide unit is present. There are, in all, 64(64 + 65) multiplication and 64 division operations used in each run of the SLI sub-program. The processing time savings are 515 microseconds for each multiplication and approximately 600 microseconds for each divide. Thus the time saved would be

$$t = 64(64 + 65)515 \times 10^{-6} + (64)600 \times 10^{-6}$$

$$\therefore \quad t = 4.25 + 0.04 = 4.29 \text{ seconds} \qquad 9.12$$

Thus the SLI process would now take less than 8 seconds.

In Chapter 8 it was shown that the time over which the input signal must be considered stationary is dependent on the background activity. From equations 8.8 and 8.10 this time is given by

$$T_{STAT} \geqslant 2\ [NT + T_A + T_{ID}] \qquad 9.13$$

Where N is the number of samples required to provide a good estimate of the autocorrelation, T is the sampling period, T_A is the time required to form the autocorrelation after enough samples have been read and T_{ID} is the time taken by the SLI process. In the generalised signal detector prototype T_A will be of the order of less than a few milliseconds, as it consists of a simple scaling routine. Thus as an approximation

$$T_{STAT} \geqslant 2\ [NT + T_{ID}] \qquad 9.14$$

We have seen that if the number of filter coefficients, L, is 64, that T = 44mS and T_{ID} is 12S. It is unlikely that less than 512 samples will provide a reasonable estimate of the filter parameters, and as a consequence (owing to the way the autocorrelation is implemented) N will be 512 + 64 = 576. It follows that

$$T_{STAT} \geqslant 2\ [(576 \times 44 \times 10^{-3}) + 12]$$

$$\therefore \quad T_{STAT} \geqslant 74.7 \text{ seconds} \qquad 9.15$$

With the inclusion of hardware multiply/divide this becomes

$$T_{STAT} \geqslant 42.5 \text{ seconds} \qquad 9.16$$

These times are unsatisfactory, and an alternative strategy is discussed in Chapter 11.

9.5 COMMENTS

The generalised signal detector prototype may be used in real-time measurement problems where, for a filter with 64 coefficients, there are no frequencies present above 23Hz and the noise may be assumed to be stationary over a period in excess of 74.7 seconds. The inclusion of a hardware multiply/divide unit would enable its application to problems where frequencies up to 45Hz are present and the noise may be considered stationary over a period of 42.5 seconds. If the number of filter coefficients is reduced then the situation improves. For example, with a 10 coefficient filter signals with frequencies up to 70 Hz are permissable if the noise may be assumed stationary over a period of 12 seconds (250Hz and 4.8 seconds respectively with hardware multiply/divide). In many ways the performance of the prototype is unacceptable, as it is applicable only to problems with low frequency components which must be stationary over long periods. The real-time processing capability of the prototype is a function of the processing power of the F100-L microprocessor, and therefore a gain in performance may be obtained by using a more powerful device. However, T_{STAT} (the minimum time over which the noise must be considered stationary) is determined mainly by the time required to collect sufficient samples for an adequate estimate of the autocorrelation to be formed and a significant improvement in performance here may only be obtained by employing a different strategy. These points are discussed further in Chapter 11.

10. RESULTS

10.1 INTRODUCTION

The work contained in this thesis has been presented in two parts; one concerned with the development of an advanced portable measurement system containing a non-volatile backing store and the other discusses an application of such a system to a class of signal detection problems. It is natural, therefore, to divide the results into two parts. Section 10.2 and its subsections contain the results of extensive tests on the magnetic bubble memory used as the non-volatile backing store. Section 10.3 examines the efficacy of the generalised signal detector as applied to the problem of the detection of the epileptic precursor in human EEGs. Both sections are terminated by a conclusion relevant to that part of the work covered in each section. Finally, section 10.4 considers the interdependence of the two parts.

10.2.RESULTS FROM PART 1 : THE MAGNETIC BUBBLE MEMORY TESTS

The MBM unit is comprised of three circuit boards, two of which are supplied by the manufacturer and one which was designed and built as an interface to a F100-L microprocessor system. The two boards supplied by the manufacturer are a memory board and a control board. Commands and data are passed by the interface program to the control board which in turn generates the signals required to read or write data to the memory board. However, the control board is designed to be included in a Texas Instruments TMS 990 microcomputer system and thus it is necessary to build an interface unit if the MBM is to be used on a different microprocessor system. The interface circuit must therefore provide a mapping from the asynchronous, multiplexed data and address bus of the F100-L to the synchronous, separate data and address busses of the TMS 990. The circuit for this, and the circuits for the control and memory boards, are given in Appendix I.

Once the interface hardware was completed it was then necessary to write the MBM driving software. There are three main functions to be implemented; initialise the controller, write a page of data and read a page of data. These functions were extremely simple to implement consisting, as they do, of a series of read/write transfers to and from the controller status, command and data words. With these routines written and tested an MBM test program could

be written, the details of which are given below. The flow charts for the MBM driving routines and a listing of the MBM test program are given in Appendix II.

10.2.1. The MBM Test Program

The objective of the MBM test program is to establish the frequency and types of error which occur when data are store in and retrieved from an MBM. This is achieved by writing known data to the MBM, reading to back and comparing the read and written data. The question is, what form should this data take? The problem is one similar to that encountered by the manufacturers of high capacity RAMs in that the total number of bit patterns possible is extremely high; in the case of the MBM there are more than $2^{92,000}$ possible patterns! Obviously, exhaustive tests for pattern sensitivity are impossible but it is possible to make selective tests on those patterns most likely to give errors. To see what these may be it is necessary to consider the internal structure of the MBM.

In the MBM the permalloy patterns under which the minute magnetic domains representing data are formed are arranged in a major-minor loop structure. Thus each of the minor loops represents a single bit in a page of data. Consecutive pages are represented by consecutive bits in the minor loops whereas consecutive bits in a page are derived from the data stored in adjacent minor loops. Thus in many ways there is a closer relationship between the data forming a particular bit position in each of the pages than there is between the bits which comprise a single page. For this reason the MBM test program was designed in such a way as to allow the user to set up different data patterns for odd and even numbered pages. The bit pattern for each page is set up as an 8 bit string which is then repeated until each of the 144 bits in the page are defined.

10.2.2. The Types And Likely Causes Of Errors

Although there may be errors generated within the MBM there are good reasons to suspect that a majority of errors would be caused by events external to the storage device. The first critical point is where data is read from the MBM. The magnetic bubble sensing system consists of a magneto-resistive element through which a constant current is passed. Should a bubble be present the change in resistance provides a voltage change across this element which is of the order of about 5mV; this must

be considered a liability if the MBM is to be used in an electrically hostile environment. From this point on, and in the data write circuitry, all signals are TTL compatible and therefore it is reasonable to expect as many errors in these parts of the circuit as in any system which consists of standard TTL units; i.e. few and far between. However, at one point in the test rig the TTL units used are pushed slightly beyond the limits of recommended operation, and this occurs between the interface and control circuits boards. These two boards are linked by a ribbon cable of a length of approximately 0.4m, which is over twice the recommended driving distance of standard TTL. The effect of this will be shown later. Any errors occuring which are attributable to a fault in the interface design will not be considered as an MBM error, but there are a number of types of error which may have occurred in either the MBM unit or the interface. These errors will be considered as MBM faults. The types of errors likely to occur are given below:

1. Single bit, independently occuring - the source of error may be either the interface or the MBM. They will, however, be considered as MBM errors.

2. Multi-bit burst, independently occuring - almost certainly MBM error unless whole bytes are affected rather than parts of bytes.

3. Single bit errors in same position in each page - MBM "hard" error, i.e. a complete failure of a minor loop.

4. Single bit errors in same position in each byte - interface failure or "hard" error in MBM controller chip, the latter being less likely.

5. Whole page error - MBM error caused by the loss of XFER IN or XFER OUT signal between controller and MBM.

6. Incorrect page read/written - address error occuring in interface unless error propogates throughout the MBM, when it is likely that a cycle has been missed between the controller and the MBM.

7. Total loss of storage - complete interface break-down (which may be tested), "hard" failure of MBM major loop or, more likely, a loss of the enable signal for the MBM drivers from the controller.

The test program runs until there have been 32768 total read/writes to the MBM or until 200 errors have been detected, whichever occurs first. The user may interrupt the program by pressing the "BREAK" key on the terminal and then typing "SUSPEND" followed by "RUN". The program may be restarted at the point of the break-in if required. A complete test run of 32768 cycles takes approximately 34 hours. The results of test runs in a laboratory environment are given below.

10.2.3. All Zeros Test Patterns

As a zero is the inactive state of a bit in the MBM (i.e. it is indicated by the lack of a magnetic bubble) an all zero pattern could not be expected to indicate an error caused within the MBM. However it is possible to read a recorded zero as a one if an error occurs during either the write or read operations. It can be seen, therefore, that the only types of error which the all zeros test pattern might reasonably be expected to detect would be types 1 and 2 as mentioned above; i.e. single and multi-bit errors which occur independently. As it is reasonable to expect these types of errors to be prevalent, though, the all zeros pattern is a valid point at which to start the test of the MBM. Type 4 errors may also be indicated, but these are, generally speaking, interface errors. The results of the all zero run are given in table 10.1.

PASS PAGE PATTERN

TESTED

TABLE 10.1 RESULT OF ALL ZEROS TEST

The number of bit transfers required for this test if 6×10^9, which is approaching the manufacturers estimated maximum soft error rate of 1 in 10^{10}.

10.2.4. All Ones Test Patterns

As one would expect, the all ones test pattern may be used to detect the same types of error as the all zeros pattern. However, as a one is an "active" state of the MBM the all ones pattern will also detect failures within the MBM. The results of the all ones test are given in table 10.2.

PASS PAGE PATTERN

TESTED

TABLE 10.2 RESULT OF ALL ONES TEST

Once again, the number of bit transfers required for this test is 6×10^9.

10.2.5. "Checherboard" Test Patterns

The "checkerboard" test pattern is frequently used to test semiconductor RAMs, but in fact it is probably more applicable to the testing of MBMs. The name of the pattern is derived from the similarity between the arrangement of ones and zeros within the device and that of the black and white squares of a chessboard. Thus within the MBM a checkerboard pattern would give alternating ones and zeros around each minor loop and, when a page is read or written, alternating ones and zeros around part of the major loop (the part of the major loop in which the data is to be found depends upon the page being accessed). In fact, this is not quite the case, as there are dummy bits (set to zero) between the data bits in the major loop so that the data bits physically align themselves with the minor loops. However, the all ones pattern mentioned above will give a true alternating one/zero pattern in the major loop, so this part of the MBM has already been tested in this way.

With the checkerboard test pattern each bit position within the minor loops alternates between the one and zero states at the maximum rate allowed (i.e. once every 10uSs) every time an access is made. It was felt that this would be more likely to produce errors than either the all ones or all zeros patterns and therefore two test runs were made using the checkerboard pattern, the results of which are given in tables 10.3 and 10.4.

PASS PAGE PATTERN

TESTED

TABLE 10.3 RESULT OF CHECKERBOARD TEST 1.

PASS	PAGE	PATTERN
7984	0180	11111111
7984	0180	11111111
7984	0180	11111111
7984	0180	11111111
7984	0180	11111111
7984	0180	11111111
7984	0180	11111111
7984	0180	11111111
7984	0180	11111111
7984	0180	11111111
7984	0180	11111111
7984	0180	11111111
7984	0180	11111111
7984	0180	11111111
7984	0180	11111111
7984	0180	11111111
7984	0180	11111111
7984	0180	11111111

TESTED

TABLE 10.4 RESULT OF CHECKERBOARD TEST 2.

The total number of bit transfers for both test runs using the checkboard pattern is 1.2×10^{10}. The error found in the second run is a whole page error, and may be due to either of the causes given for error types 5 and 6 as shown above. However, as each bit must have been inverted to produce the "all-ones" error pattern the error cannot be caused by a loss of either the XFER IN or XFER OUT signals as this would lead to the read page being set to all zeros. Thus those bits in the original, written page which were zero would not show up as errors. It follows that the error found in checkerboard test 2 must be a type 6 error; that is an addressing error occuring within the interface, and therefore the MBM may be considered free of faults over all the runs.

The likely cause of the addressing error found in checkerboard test run 2 is the operation of the address driving logic above the recommended limits for TTL devices (see section 10.2.2.). Such errors would not occur in the final system where the interface board would be situated physically closer to the MBM control board.

10.2.6. Conclusions (MBM)

After a total of 2.4×10^{10} bit transfers (or 136 hours of test runs) using different test patterns no errors were found which could be attributed to a failure of the MBM. This result is consistent with the manufacturers estimate of a soft error rate of 1 bit in 10^{10} bit transfers. However, the total duration of the test runs cannot be considered sufficient for the runs to prove conclusively the reliability of MBM technology. Furthermore, the tests were performed on only one device and it could not be claimed that the results quoted are in any way average for this type of memory. Such results could only be gained by testing a number of devices over long periods. Nonetheless, the initial results are promising and, in the present author's opinion, indicate the suitability of MBM for use as the non-volatile storage medium in portable instruments.

The experiments on the MBM were conducted in a laboratory environment, but it was stated above that the MBM may be prone to errors if it was operated in an electrically hostile environment. After much consideration it was decided that running the MBM tests in a "controlled" electrically hostile environment would not provide conclusive evidence as to the type of errors which might be induced by the noise. For example, a sharp spike in the background noise might induce a single bit error whereas a burst of noise could induce a multi-bit error. It follows that experiments designed to test the MBM in such environments must use a pre-conveived idea of the qualities of the noise. An ad hoc experiment was tried, however, whereby an old electric drill was operated in close proximity to the MBM while a test run was in progress. The failure of this method to induce any errors added further weight to the conviction that a controlled experiment would not provide a useful result. This does not mean that a user who can determine the type of noise in which an MBM may be required to operate in should not first establish, by experiment, the likely types of error which may occur. Certainly, the facilities now exist for performing just such an experiment.

10.3 RESULTS FROM PART 2 : THE GENERALISED SIGNAL DETECTOR

The generalised signal detector consists of two processes; the identification of the matched filter and its implementation for the solution of a signal detection problem. It follows that the performance of the system as a signal detector is dependent upon two things. The first concerns the identification process, for if the SLI technique can provide only a crude estimate of the filter coefficients then the performance of the matched filter may be seriously affected. Assuming, however, that the identification procedure performs its task to an acceptable degree it may still be that a matched filter will not provide an acceptable solution. The results given below refer to the two parts of the generalised signal detector as separate entities, as the effects of any interaction between the two processes (which will influence factors such as the maximum sample rate and the "speed" of any non-stationarities allowed) are discussed in Chapters 9 and 11.

10.3.1. Results Pertaining To The Identification Process

The ability of the SLI technique to track changes in the background noise present in a signal detection problem depends upon the speed of the convergence from an old, suboptimal estimate of the matched filter to the new estimate based upon the measured autocorrelation of the noise. The two factors affecting the rate of convergence are the difference between the two estimates of the filters and the value of the error correction coefficient, α. Of these the former will be largely dependent upon the application, as large changes in the quality of the background noise will necessitate a correspondingly large change in the filter coefficients. Thus experiments were conducted to observe the effects of choosing various values for α.

The literature on SLI is not particularly informative as to the selection of a value for α, recommending only that it should be less than 1 for fast convergence and much less than 1 for a small steady-state error (33). As a steady-state situation does not arise in the application of SLI in the generalised signal detector (as the background noise is assumed to be non-stationary) it follows that the only relevant guideline for the choice of α is that it should be less than 1. Furthermore, it may be argued that the suggestion that α should be reduced so that only a small steady-state error is incurred is not valid. The SLI technique works by adding an error correcting vector to the present estimate of the filter. The length

of this vector is not decided by the value of α alone, but also by the vector $(y_{n-1}-\hat{y}_{n-1})$; that is, the difference between the true and measured output of the optimal filter. Therefore, as the estimate of the filter approaches the true filter the vector length will decrease even if α is kept constant. What is more, if α is decreased too far it is possible for the error correcting vector length (in a digital system) to be less than that which may be contained within the least significant bit of the computer word, so that further convergence is impossible. Consideration of this argument led to the conclusion that it would not be a secure strategy to vary α throughout the course of a measurement so as to maximise the rate of convergence when the error in the filter was large and minimise the error at other times. A better solution would be to find a compromise value for α whereby an acceptable rate of convergence might be found without incurring a lack of accuracy in the final estimate.

Two experiments were devised to obtain empirically an acceptable value for α. The first assumed the background noise to be ideally white by using a preset autocorrelation where Rnn(0) was set to a large value and Rnn(1) to Rnn(63) were set to a constant, smaller value. The impulse response of the matched filter was preset to a constant value so that the initial estimate of the filter coefficients was poor. This enabled a comparison of the various rates of convergence. As the number of iterations increased the estimate of the filter improved, allowing differences in the final estimate of the filter to be observed. The value of α is contained in location 296 (OCT 450) as a 16 bit fixed point fraction, and this may be changed by the use of the control handset of the F100-L DOS development system. The values chosen for α were $1-2^{-15}$, 0.5, 0.25, 0.125 and 0.0625, and the impulse responses obtained after 1 iteration are shown graphically in figure 10.1.

After the first iteration the closest approximation (in terms of the amplitude of the coefficients) occurs, not surprisingly, when α is comparatively large, i.e. when $\alpha = 1-2^{-15}$. However, as the filter coefficients are normalised before the filter is implemented in order to take the maximum advantage of the computer word it is the <u>shape</u> of the estimate which is important. On this basis the best estimate of the matched filter is obtained when $\alpha = 0.0625$, the smallest value used in the experiment. The implication is that α should be small in order that the SLI may provide a good approximation after only 1 iteration. There will,

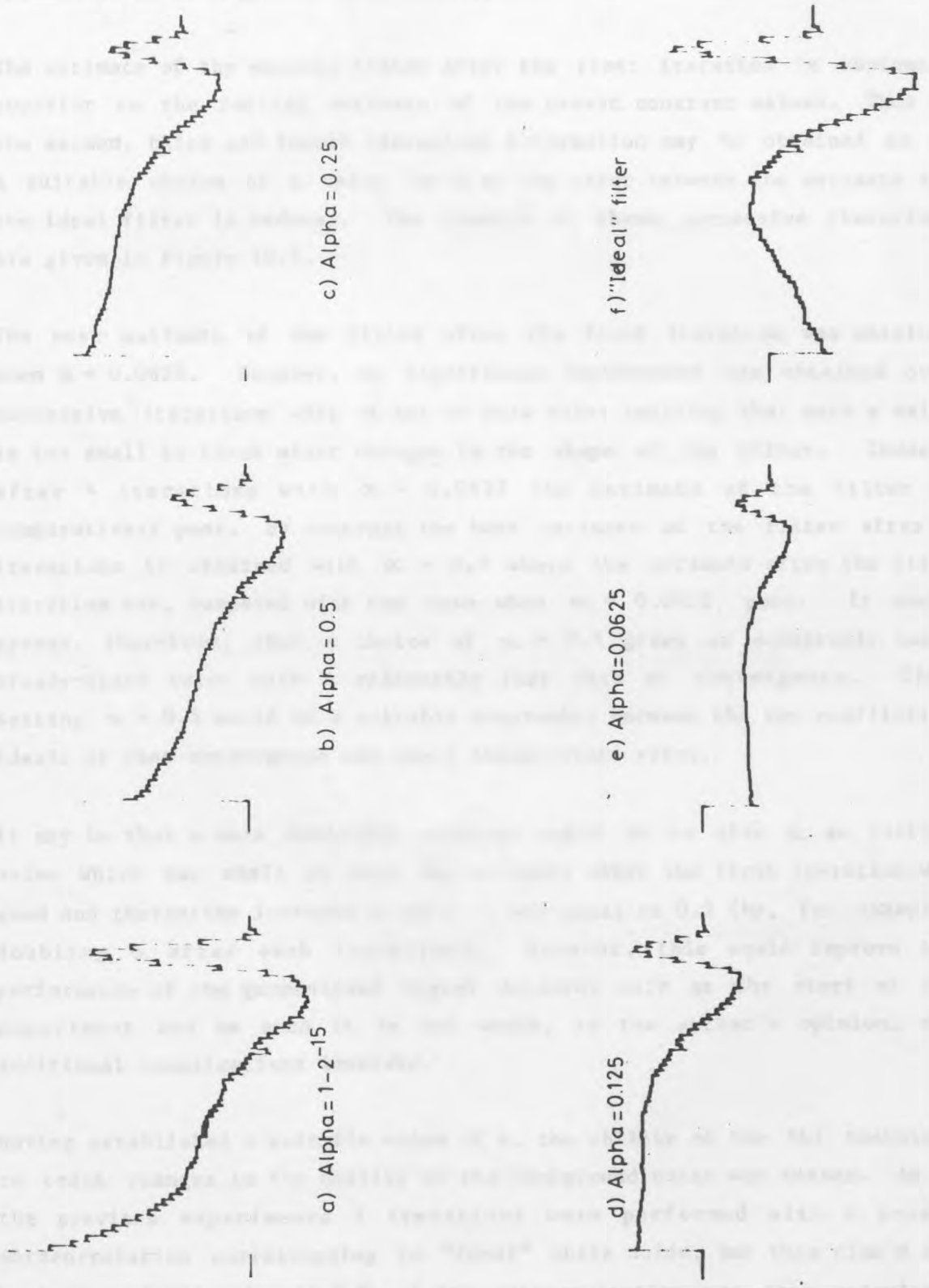


FIGURE 10.1 MATCHED FILTER ESTIMATES AFTER ONE ITERATION

of course, be a limit as to how small α can be before either underflow in the calculations or excessive quantisation occurs and there may well be an optimum value of α greater than this limit.

The estimate of the matched filter after the first iteration is obviously superior to the initial estimate of the preset constant values. Thus on the second, third and fourth iterations information may be obtained as to a suitable choice of a value for α as the error between the estimate and the ideal filter is reduced. The results of these successive iterations are given in figure 10.2.

The best estimate of the filter after the first iteration was obtained when $\alpha = 0.0625$. However, no significant improvement was obtained over successive iterations with α set to this value implying that such a value is too small to track minor changes in the shape of the filter. Indeed, after 4 iterations with $\alpha = 0.0625$ the estimate of the filter is comparatively poor. By contrast the best estimate of the filter after 4 iterations is obtained with $\alpha = 0.5$ where the estimate after the first iteration was, compared with the case when $\alpha = 0.0625$, poor. It would appear, therefore, that a choice of $\alpha = 0.5$ gives an acceptably small steady-state error with a reasonably fast rate of convergence. Thus setting $\alpha = 0.5$ would be a suitable compromise between the two conflicting ideals of fast convergence and small steady-state error.

It may be that a more desirable strategy would be to give α an initial value which was small so that the estimate after the first iteration was good and thereafter increase α until it was equal to 0.5 (by, for example, doubling α after each iteration). However, this would improve the performance of the generalised signal detector only at the start of the experiment and as such it is not worth, in the author's opinion, the additional complications involved.

Having established a suitable value of α, the ability of the SLI technique to track changes in the quality of the background noise was tested. As in the previous experiments 4 iterations were performed with a preset autocorrelation corresponding to "ideal" white noise, but this time α was kept at a single value of 0.5. A new autocorrelation was then calculated from an input derived from a noise generator passed through a low-pass filter with a cut-off frequency of 10Hz. A further 4 iterations were performed in order to examine the ability of the system to respond to

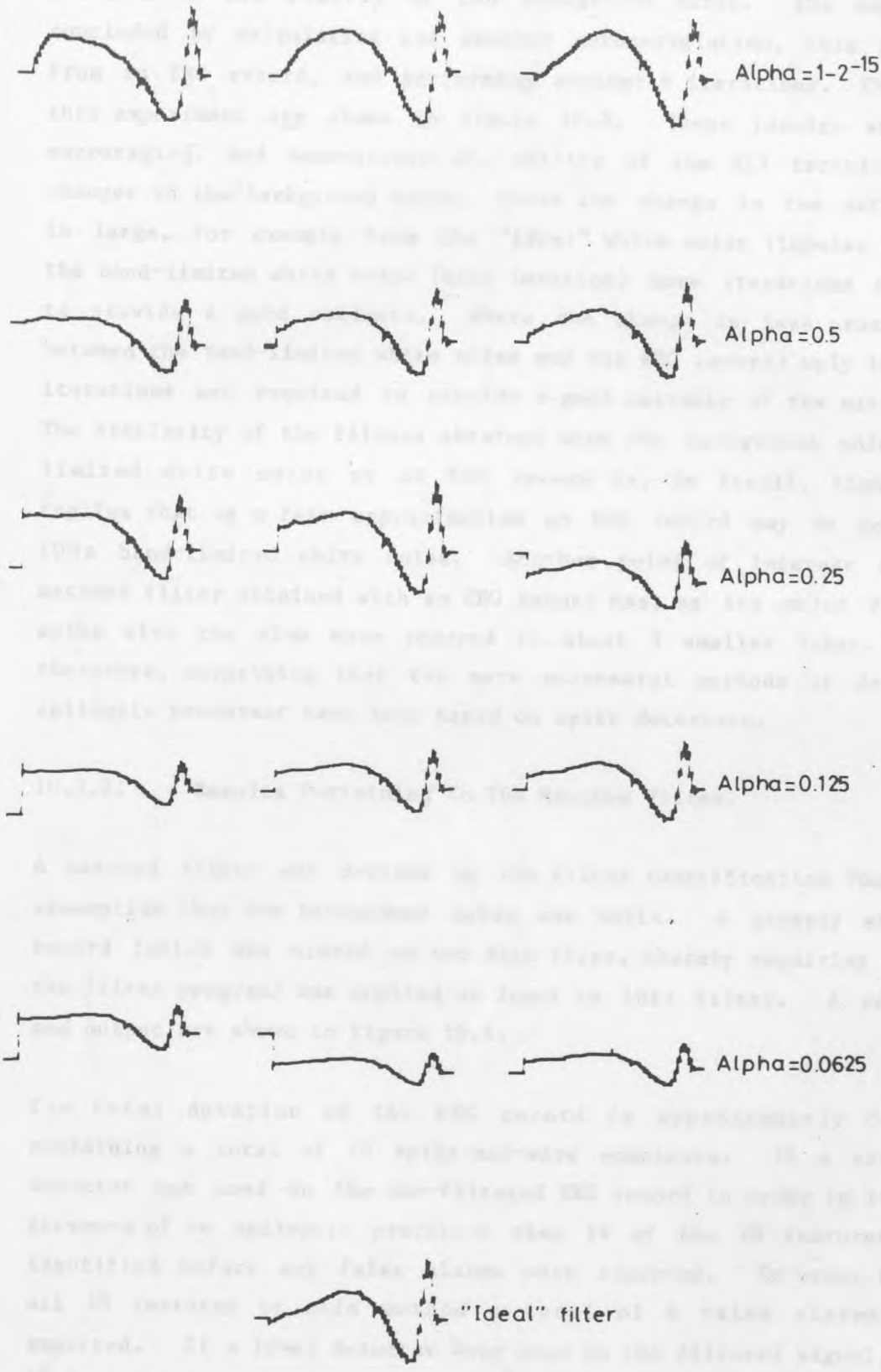


FIGURE 10.2 MATCHED FILTER ESTIMATES AFTER THE $2^{nd}, 3^{rd}$ AND 4^{th} ITERATIONS

changes in the quality of the background noise. The experiment was concluded by calculating yet another autocorrelation, this time derived from an EEG record, and performing another 4 iterations. The results of this experiment are shown in figure 10.3. These results are extremely encouraging, and demonstrate the ability of the SLI technique to track changes in the background noise. Where the change in the autocorrelation is large, for example from the "ideal" white noise (impulse function) to the band-limited white noise (sinc function) more iterations are required to provide a good estimate. Where the change is less pronounced (i.e. between the band-limited white noise and the EEG record) only two or three iterations are required to provide a good estimate of the matched filter. The similarity of the filters obtained when the background noise is band-limited white noise or an EEG record is, in itself, significant and implies that as a fair approximation an EEG record may be considered as 10Hz band-limited white noise. Another point of interest is that the matched filter obtained with an EEG record has, as its major feature, the spike with the slow wave reduced to about 3 smaller lobes. It is not, therefore, surprising that the more successful methods of detecting the epileptic precursor have been based on spike detectors.

10.3.2. Results Pertaining To The Matched Filter.

A matched filter was derived by the filter identification routine on the assumption that the background noise was white. A grossly abnormal EEG record (which was stored on two disk files, thereby requiring two runs of the filter program) was applied as input to this filter. A sample input and output are shown in figure 10.4.

The total duration of the EEG record is approximately 64 seconds, containing a total of 18 spike-and-wave complexes. If a simple level detector was used on the non-filtered EEG record in order to indicate the presence of an epileptic precursor then 14 of the 18 features could be identified before any false alarms were incurred. In order to identify all 18 features by this method a total of 6 false alarms could be expected. If a level detector were used on the filtered signal, 16 of the 18 features would be indicated without any false alarms. To detect all 18 features by this method a total of 2 false alarms would be incurred.

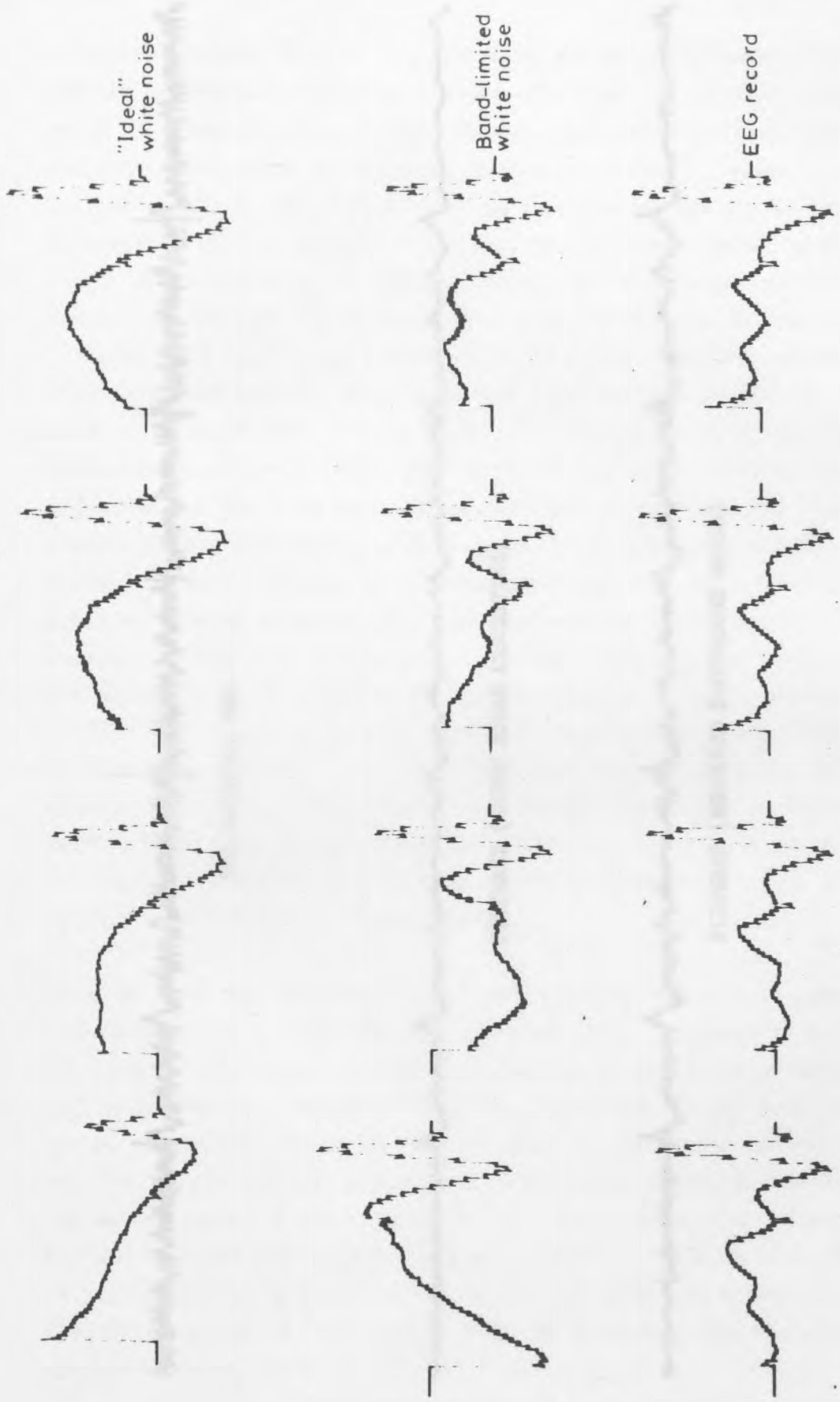


FIGURE 10.3 THE TRACKING ABILITY OF S.L.I.

FIGURE 10.4 ORIGINAL AND PROCESSED ELECTROENCEPHALOGRAMS

A second matched filter, derived from an autocorrelation calculated from the EEG record, was used to process the same EEG signal, the output of which is also shown in figure 10.4. Once again, 16 of the 18 features could be identified by a level detector without incurring any false alarms. Only 1 false alarm need be incurred before all 18 features could be idenfitied. The feature which resulted in the 1 false alarm, although not a spike-and-wave, is unusual in that it is a large amplitude slow wave similar to the wave in the epileptic precursor. It is likely that such features will always give a false reading when a matched filter is used to process an EEG record. The reason for this is that of the 64 coefficients used to define the filter 54 of them relate to the wave part of the epileptic precursor. Thus the matched filter is weighted towards the detection of the slow wave rather than the spike, implying that slow waves without spikes are likely to give a positive reading. Conversely, spikes which are not followed by a slow wave are extremely unlikely to give a positive reading so that one could expect a low false alarm rate when, for example, artefact is present in the EEG record. Slow waves pass through the matched filter seemingly unaltered, whereas a spike-and-wave complex, by contrast, gives a definite spike at the filter's output, with little or no slow wave present. It should, therefore, be possible to discrimate between a true indication of a spike-and-wave and a false slow wave reading with some extra processing. However, this is a problem peculiar to the processing of EEG records rather than general signal detection and as such will not be considered here.

In many ways the performance of the matched filter derived from the autocorrelation of the EEG was disappointing. Although it is superior to the simple white noise matched filter it does not provide as positive an indication of the presence of a spike-and-wave as one would like. There may be two reasons for this, one of which is applicable to the use of the matched filter to any problem while the other is due to the nature of the EEG record used. A matched filter is, quite simply, that filter which has the optimum defined signal-to-noise ratio. If the matched filter output is passed through a simple level detector in order to detect features then the performance of the system will be dependent on the "size" of the signal-to-noise ratio.

If the signal is large compared to the noise then the filter will perform well. If, however, the noise swamps the signal then the filter will perform badly, in which case further processing might be necessary, for example by taking the second derivative of the matched filter output to detect the sharp spike which characterises the presence of a feature. In the case of the detection of the spike-and-wave complex in an EEG record the spike is often considerably greater than the noise but the slow wave is frequently no greater in amplitude than the noise and is only distinguished by its greater duration. As the matched filter is weighted towards the detection of the slow wave it is, perhaps, surprising that it works as well as it does. In problems where the signal-to-noise ratio is higher than in the epileptic precursor detection problem one could expect, therefore, an improved performance in the generalised signal detector. Unfortunately, should the signal-to-noise ratio be smaller than that in the EEG record it is possible that the performance of the generalised signal detector would be less than adequate. In the case of the detection of the epileptic precursor the situation is made worse by the nature of the EEG record used. Over the 64 seconds duration of the record there are 18 spike-and-waves, each of which has a duration of approximately 1/3 second. It follows that nearly 10% of the EEG record is made up of spike-and-waves implying that the calculated autocorrelation is not of the background noise but of a 9:1 noise and signal mix. Thus the matched filter will optimise the signal-to-noise ratio when the signal is itself considered as part of the noise, albeit to a small extent. It must be stressed that the EEG record used is grossly abnormal and that one would not expect so many abnormalities in such a short period of time. In a more normal EEG, where the abnormalities are few and far between, one would expect a more accurate approximation to the autocorrelation of the noise, and hence a more optimal matched filter.

10.3.3. Conclusions (Generalised Signal Detector)

The ability of the SLI algorithm to track changes in the quality of the background noise enables the estimate of the matched filter coefficients to be close to that of the optimal filter provided that such changes are not excessive or occur too rapidly for the system to respond adequately. Thus the proposed generalised signal detector promises to provide an excellent solution to signal detection problems where the background noise is non-stationary. However, not all signal detection problems are able to be solved by the application of a matched filter and these problems would

require a different solution. Furthermore, as the autocorrelation of the combined signal and noise is calculated (rather than just the noise) the generalised signal detector cannot be used where a signal is frequently present.

10.4 MAGNETIC BUBBLE MEMORY AND THE GENERALISED SIGNAL DETECTOR

The advantages of magnetic bubble memory technology compared to other forms of computer storage in a portable microprocessor system are apparently, after an initial investigation, supported by a high degree of reliability. The choice of an MBM based non-volatile bulk storage unit for a portable microprocessor system would, therefore, seem reasonable.

The generalised signal detector, if implemented on a portable dual-processor system, is one of a number of applications where bulk non-volatile store is essential. In the majority of signal detection problems there will be a trade-off between false alarms and failures to detect signals which are present. This is particularly true in the case of the detection of the epileptic precursor where the generalised signal detector is prone to indicate slow waves as spike-and-wave complexes. It is not envisaged, therefore, that the generalised signal detector would operate in a stand-alone mode but instead be used as an aid for a trained observer who would make the final decision as to whether a signal was present. Thus when the system detects a signal it must store a section of the input where, according to the detection algorithm, the signal has occurred for further analysis by the trained observer. Should the experiment be conducted over a period of time where a number of signals may occur the demand for storage space will be high. Further, if (as in the case of the RSP) the system does not contain any mechanism for the displaying of results then they must be stored in a non-volatile memory until such time as the system is returned to its host, where display facilities would be provided. The need for bulk, non-volatile store is, therefore, self-evident.

11. DISCUSSION

11.1 INTRODUCTION

The work contained in this thesis forms part of a group project on the application of multi-processor systems to measurement problems. As a first step to obtaining a clearer picture of the problems in the use of multi-processor configurations in instruments a prototype dual-processor system was developed and examined in detail in a project running in parallel to the work contained here (Mainwaring-Samwell, ref 34). The intention was to provide a dual-processor based portable instrument complete with a suitable application. The generalised signal detector, which forms the bulk of the present author's work, is just such an application. The provision of a bulk non-volatile store for a portable measurement system (as described in Part 1 of this thesis) may be considered as an element of either of the two parts of the overall project (development and application) as it is essential to both.

Unfortunately, developing a system and providing it with an application concurrently implies that, for some time, the application will not have a system on which it may be run. This is true of the generalised signal detector, whose concept and development were proceeding at a time when either there was no dual-processor system available or when the prototype system was undergoing extensive checks and measurements. It was necessary, therefore, to develop the generalised signal detector on a single processor system (see Chapter 9). It follows that the final performance of the signal detector, which will be influenced by the efficiency of the dual-processor and its resource handling, may only be estimated at the time of writing.

11.2 PERFORMANCE DEGRADATION IN A DUAL PROCESSOR

In a dual-processor system the amount of processing power available for application to a given task will inevitably be less than the sum of the processing powers of the two processors. This shortfall (which we shall call degradation) occurs because of the need to protect shared resources from inadvertant data corruption and to avoid situations such as the "deadly embrace" (34). Apart from the processing time which must be dedicated to such housekeeping tasks there will also be some time lost through the two processors contending for a shared resource. It follows that the amount of degradation will depend upon the number of accesses the processors make to shared resources (coupling) and how often these accesses result in contention. One may see that

it is extremely difficult to provide an estimate of the expected degradation in a dual-processor system, for although the number of accesses made to the shared resources is usually known it may well be impossible to establish how many of these will result in contentions. One meaningful way of providing estimates of the degradation is by forming models of "typical" dual-processor applications. It is then possible to establish the percentage degradation (percentage, that is, of the processing power of one processor) for varying degrees of processor coupling by the use of non-invasive measurement techniques. This analysis has been performed for a dual processor F100-L system by Mainwaring-Samwell (34), the results of which are applied below to the problem of degradation in the generalised signal detector when implemented on such a system.

The case of the dual-processor F100-L system as implemented for this research is unusual in that any accesses to the shared resource (which is a block of common store) made by the main processor will <u>always</u> result in a form of contention. To see why this is so we must examine the structure of the system.

The configuration used in the dual F100-L system consists of an I/O processor and a main processor, each of which has its own private store. Furthermore, the input/ouput processor also has on its data/address bus all the system peripherals and the shared resource; a block of common store which may also be accessed by the main processor via DMA transfers through the "back-to-back" interface set (refer to figure 4.3). It follows that when the main processor attempts to access the common memory that

1. The back-to-back interface set must wait until the I/O processor's bus is free before it may initiate a DMA cycle.

2. When the back-to-back interface does control the bus the I/O processor must wait until the transfer is complete before it can access <u>any</u> of the devices on the bus, shared or private.

An alternative strategy would be to place the common store on a separate data/address bus where it may be accessed by DMA transfers by both processors, as in figure 11.1. Such a configuration would avoid accesses by the main processor to the common store affecting the performance of the I/O processor apart from when both processors attempt to gain control of the shared resource. Unfortunately there is a practical reason why such a dual F100-L configuration would not necessarily be an improvement over the structure used

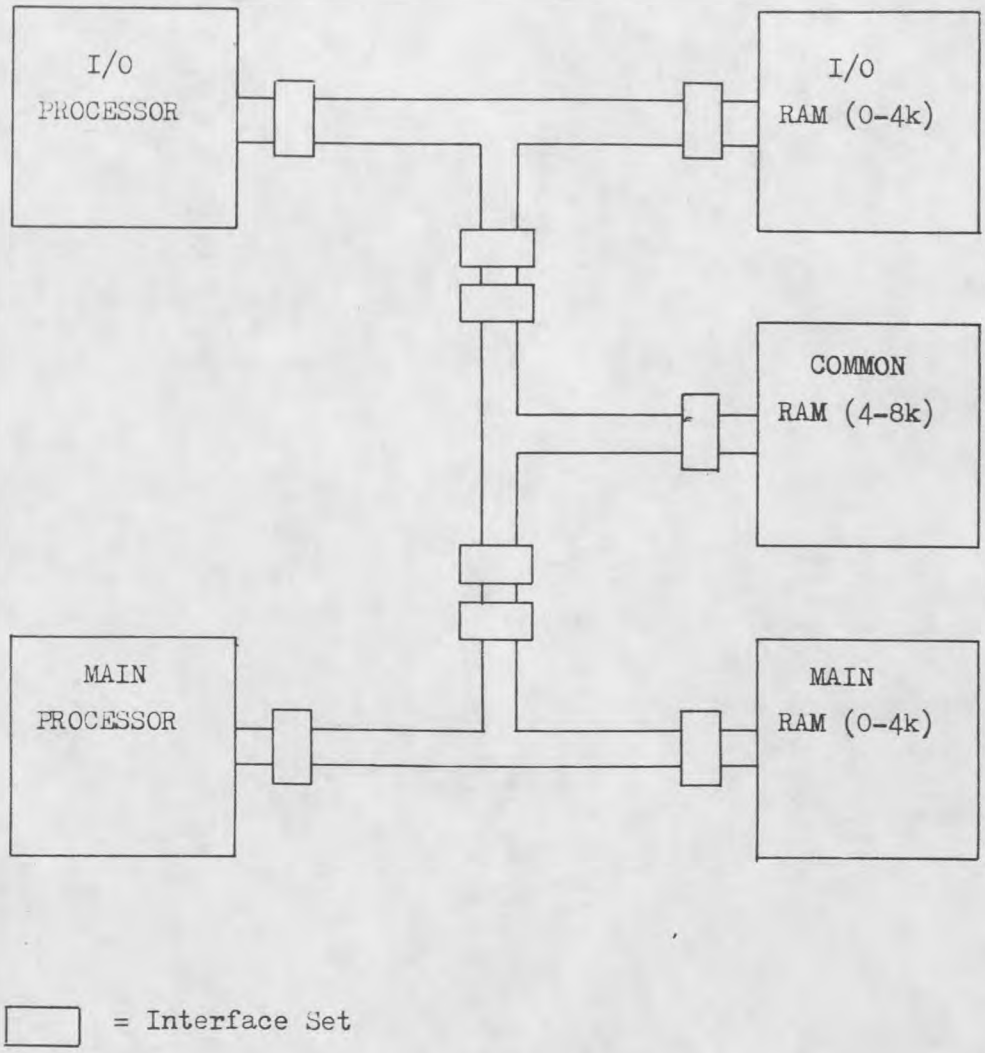


FIGURE 11.1 ALTERNATIVE DUAL PROCESSOR CONFIGURATION WITH COMMON RAM ON A SEPARATE BUS

in the prototype. The results obtained by Mainwaring-Samwell (34) indicate that, in the F100-L system, a major cause of the loss of processing power is the excessively long propagation delays through the back-to-back interface set. In the configuration shown in figure 11.1 the main processor would have to access the common store via a back-to-back interface set rather than by a direct access as in the prototype. It is likely that the extra delay incurred by the inclusion of a further back-to-back interface set would degrade the performance of the I/O processor far more than the DMA transfers initiated by the main processor.

Performance figures for the generalised signal detector when it is implemented on a single processor system are given in Chapter 9. However, in the final instrument a dual-processor configuration is proposed. It would be desirable, therefore, to provide an estimate of the degradation in performance when the generalised signal detector is run on the prototype dual-processor F100-L system. To do this it is necessary to examine the flow of data through the system when the generalised signal detector is in operation, and thereby deduce the degree of coupling between the two processors and to see whether the data flow model compares with one of those used by Mainwaring-Samwell as examples of typical dual-processor applications. As the I/O processor has two separate tasks - the calculation of the autocorrelation and the filter identification program - it follows that there will be two data flow models depending upon which of the two tasks is being performed. The two cases are examined below.

11.2.1. Processor Coupling During Autocorrelation

There are a total of four separate processes in the generalised signal detector. Three of these - the input, output and autocorrelation processes - run on the I/O processor and therefore they will not compete with each other for control of the shared resource. The fourth process, which is the filter program, may compete against any of the others for the shared resource.

Each time a new sample is read by the input process it, and the previous 63 samples (in our example), must be made available to both the autocorrelation and filter processes. In addition the filter process must make its output sample available to the output process. The flow of data through the system is shown in figure 11.2. Note that the filter coefficients are held in common store, as it is necessary for the I/O processor to be able to access them when the filter identification process

is running. It can be seen that the total number of accesses made to the common store is 195; 66 by the I/O processor and 129 by the main processor. With hardware multiplication on the F100-L the estimated sample period is 11mS (see Chapter 9). This implies that the main processor accesses the common store, on average, once every 85 μS and the I/O processor accesses common store once every 170μS. Obviously, the two processors are extremely closely coupled and one would expect the degradation to be quite high.

Most of the accesses to the common store are made to the input buffer so that the autocorrelation and filter processes may fetch the latest 64 samples. An alternative strategy would be to provide each of these processes with their own private buffer, thereby reducing the number of accesses made to the common store. The number of accesses made to the common store each time a new sample is read is now reduced to 68; 2 by the I/O processor and 66 by the main processor. Furthermore, this new arrangement corresponds very closely to one of the data flow models examined in detail by Mainwaring-Samwell, so that estimated degradation figures are available.

Model 2, as proposed by Mainwaring-Samwell in reference 34, consists of both processors writing 1 sample and reading 1 sample to and from the common store and then performing a program loop which may be of varying durations. Altering the length of the program loops enables the user to set the degree of coupling of the two processors. Now, it may appear that this model does not adequately represent the flow of data and the resource contention that occurs within the generalised signal detector. However, as it is the filter process which determines the sample rate it follows that the two processors will only contend for control of the common store around the time when a new sample is to be read. That is, there will only be a conflict when samples are transferred between the two processors. The accesses made to the common store by the main processor in order to fetch the filter coefficients will not result in contention. Furthermore, it is not necessary to protect that part of the common store which holds the filter coefficients with semaphores as it is permissable to alter any of the coefficients at any time. There will not be, therefore, any degradation within the main processor due to semaphore handling apart from that required when samples are being transferred between the processors

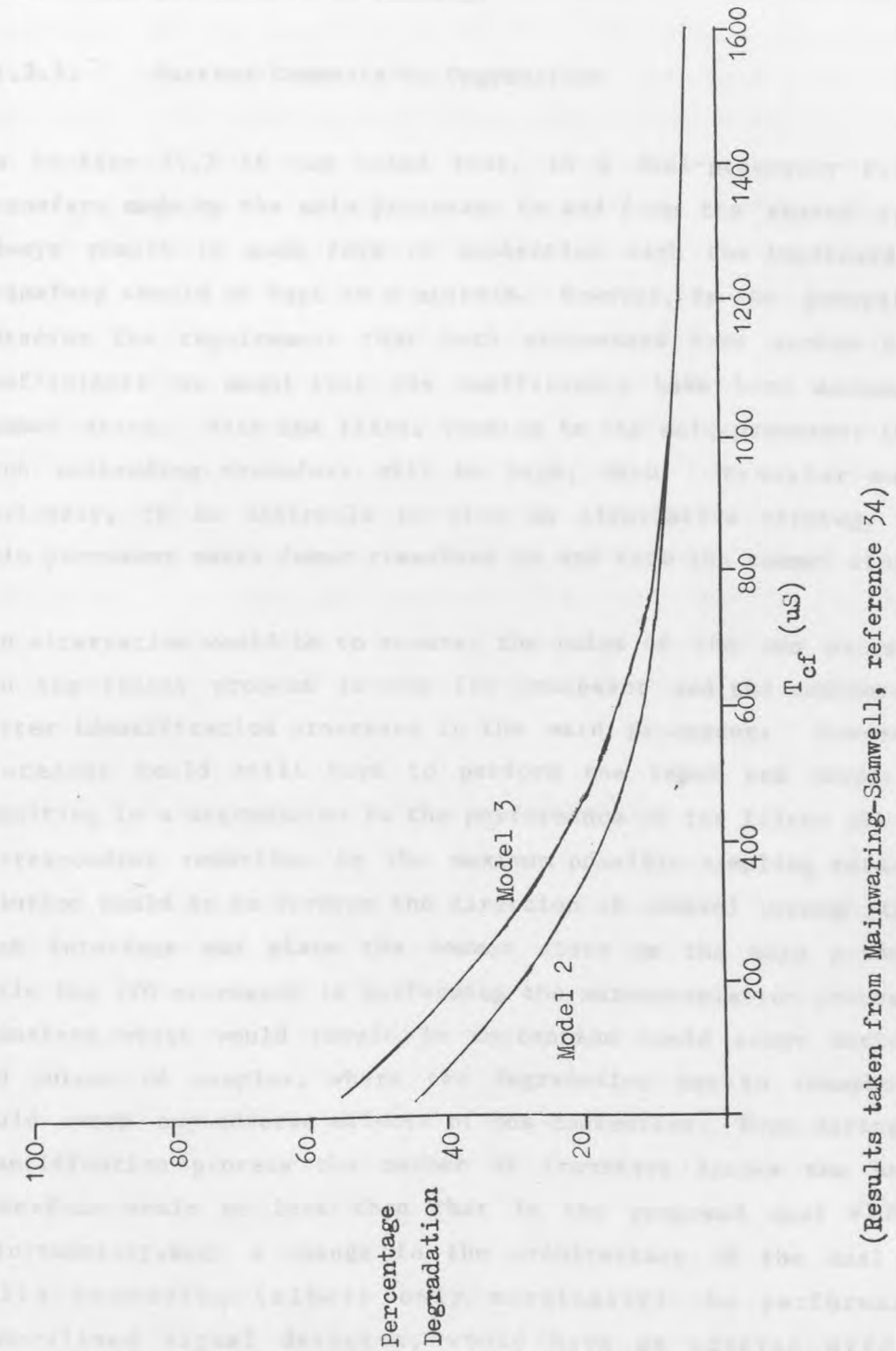
(there will be a minimal amount of degradation owing to the fact that the common store, as far as the main processor is concerned, has a longer access time than its own store).

The above argument indicates that the generalised signal detector may, as a good approximation, be equated with the proposed Model 2 for the purposes of examining the expected degradation in the dual processor. Figure 11.3 shows the measured percentage degradation against T_{CF}, the duration of the loop in the main processor. In the case of the generalised signal detector T_{CF} will be the sample period, which is 11mS. The graph in figure 11.3 stops at 1.6mS, when the degradation is about 5% and levelling off. We can assume, therefore, that the degradation in the signal detector will be 5% or less.

11.2.2 Processor Coupling During Filter Identification

The essential difference in the flow of data through the dual processor signal detector when the filter identification process is running is that, unlike the autocorrelation process, the identification process does not require access to the input samples but instead it must update the filter coefficients. The fact that the identification process does not require access to the input samples does not affect the processor coupling compared to that when the autocorrelation process is running as it was shown above that it is desirable to keep the input samples for the I/O processor in private store to reduce the possibility of contention. However, the updating of the filter coefficients will increase the coupling of the two processors as there will now be some contention between the two processors for access of the coefficients.

It was indicated in the preceding sub-section that it is not necessary to protect the filter coefficients by the use of semaphores, as individual coefficients may be altered at any time to provide a better estimate of the filter (provided, of course, that the identification process is working correctly). It follows that although we may determine the degree of coupling of the two processors the actual degradation in performance will be less than that indicated in figure 11.3, where the shared resources are semaphore protected. Indeed, the contention for the filter coefficients will result only in a wait for a transfer to be completed; at worst this will result in a delay of about 1μS. Furthermore, as the identification and filter processes are not synchronised (that this to say



(Results taken from Mainwaring-Samwell, reference 34)

FIGURE 11.3 PERCENTAGE DEGRADATION IN TWO DUAL PROCESSOR CONFIGURATIONS AS CONTENTION DIMINISHES

that the two processes will, in general, require access to the coefficients at different times) the possibility of contention is not high. It is reasonable, therefore, to assume that the degradation in performance will be only slightly worse when the identification process is running than the figure of 5% obtained for the signal detector when the autocorrelation process is running.

11.2.3. Further Comments On Degradation

In Section 11.2 it was noted that, in a dual-processor F100-L system, transfers made by the main processor to and from the shared resource will always result in some form of contention with the implication that such transfers should be kept to a minimum. However, in the generalised signal detector the requirement that both processors have access to the filter coefficients has meant that the coefficients have been assumed to be in common store. With the filter running in the main processor the number of such contending transfers will be high, with 1 transfer every 170μS. Obviously, it is desirable to find an alternative strategy whereby the main processor makes fewer transfers to and from the common store.

One alternative would be to reverse the roles of the two processors, i.e. run the filter process in the I/O processor and the autocorrelation and filter identification processes in the main processor. However, the I/O processor would still have to perform the input and output processes, resulting in a degradation in the performance of the filter process with a corresponding reduction in the maximum possible sampling rate. A similar solution would be to reverse the direction of control through the back-to-back interface and place the common store on the main processor's bus. While the I/O processor is performing the autocorrelation process the only transfers which would result in contention would occur during the input and output of samples, where the degradation due to semaphore handling would swamp any adverse effects of bus contention. Even during the filter identification process the number of transfers across the back-to-back interface would be less than that in the proposed dual F100-L system. Unfortunately,such a change in the architecture of the dual processor, while improving (albeit only marginally) the performance of the generalised signal detector, would have an adverse effect on the performance of the system in other applications, particularly those which are I/O intensive.

A third alternative, which does not require any changes in the dual-processor concept, would be to provide both processors with separate versions of the filter coefficients in their private stores. At the start of each filter identification run the two versions of the coefficients would be identical. The filter would then continue to use its version while the identification process alters all the coefficients in its own version. At the completion of the identification process the main processor could be flagged to indicate that a new set of coefficients were available. The updated coefficients could then be transferred en massse, via the common store, to the main processor's private store and the procedure repeated. As the block transfer of coefficients would occur every 8 seconds (with hardware multiplication and a 64 coefficient filter) degradation would be minimal, although at least one extra instruction would be required in the filter process in order to examine the flag indicating a new set of coefficients are available. Although this flag must itself be in common store the number of transfers would be reduced to $1/64^{th}$ of its previous value, at the expense of one extra instruction and a block transfer at infrequent intervals. Whether any gained improvement in performance would justify the increased complexity of the system is a moot point. At best the improvement gained would be in the order of microseconds per sample, and even then this improved performance would be gained mainly in the I/O processor, where there is no critical real-time process running.

11.3 IMPROVING THE PERFORMANCE OF THE SIGNAL DETECTOR

The Ferranti F100-L microprocessor on which the signal detector is implemented is a comparitively old device in terms of microprocessor technology and performance. Thus the most obvious way of improving the signal detector's performance, and the one most likely to be effective, is to replace the F100-L with one of the new generation of 16-bit microprocessors. Of these it would appear that the best microprocessor in terms of raw processing power, is the Motorala M68000. What, then, would be the expected improvement in performance if the M68000 replaced the F100-L ?

The only true real-time process in the signal detector is the filter (the autocorrelation, although it requires regular input samples, does not provide a real-time output and therefore it could be implemented by storing all incoming samples and then performing the calculation in non-real-time), and as the performance of this process is critical in establishing the maximum possible

sampling rate it was the filter sub-program which has been "bench-marked". The instructions for the filter process implemented on a M68000 microprocessor are given in Appendix V. The estimated maximum sample rate if the M68000 is clocked at 4MHz is 430 Hz, giving an improvement of more than an order of magnitude over the F100-L. However, much of this improvement in performance is due to the availability of hardware multiplication in the M68000 whereas in the F100-L multiplication is performed by a subroutine. Nonetheless, if the F100-L is provided with a hardware multiply/divide unit the improvement in performance would still be of the order of 8 times.

It may be argued that bench-marking is not a fair method of establishing the relative performances of processors. While this is true, the purpose of the bench-mark programs given above is to establish the performances of two processors given a particular task and so bench-marking is, in this case, a valid technique.

The improved performance of the later generation of 16-bit microprocessors is due, to some extent, to the fact that data and instruction fetches are performed while the processor is performing the instruction decoding and implementation. As a consequence the processor's bus is in constant use. This has a serious effect on the concept of DMA transfers; a technique originally devised to enable peripherals to transfer data to and from store at those times when the bus was free so that little or no processing time was lost. It follows that with the newer processors a large number of DMA transfers could seriously degrade the performance of the processors. In the case of multi-processor systems, whereby data is transferred between the processors by DMA, the percentage degradation due to the inclusion of the extra processors will be much higher than that incurred in older systems. However, as the processing power of later microprocessors is so much greater an increased percentage degradation may be tolerated as the overall performance would still be greater than with processors such as the F100-L.

11.4 IMPROVING THE FILTER IDENTIFICATION

The sequential learning identification (SLI) technique used to determine the matched filter's coefficients has proved to be adequate, but there are a number of problems associated with it. By far the greatest problem is the large delay between the first calculations for the autocorrelation of the noise and a final estimate of the filter coefficients. To overcome this problem there are two alternatives; replace the SLI with another strategy or find a way to improve

the SLI performance. As was shown in Section 11.2 the performance of a program may be greatly improved by changing the processor to one of greater power. However, in the case of SLI the greatest improvement in performance would be gained if there were some way of choosing an optimum value of the error correcting coefficient such that the number of iterations required to provide a reasonable estimate of the filter coefficients is kept to a minimum. The only guideline available for the selection of the error correcting coefficient states that it must be less than unity (much less than unity for a small steady-state error) which is obviously not satisfactory. The experimental results quoted in Chapter 10 indicate that a choice of a value of 0.5 appears reasonable, although this is dependent on the quality of the present estimate.

The alternative strategies to SLI were covered in Chapter 7 and include direct deconvolution (via a FFT), optimisation methods and regression techniques. Of these, optimisation methods must be discounted as a real alternative to SLI owing to the unpredictable convergence characteristics of the more efficient methods, the large amount of processing time required and the possibility of converging to a local minimum if one cannot guarantee unimodality in the target function. However, both direct deconvolution and regression methods (in particular sequential regression) are valid alternatives to SLI. The increased complexity of integrated circuits has led to the greater processing power of microprocessors, as was shown in Section 11.2 As both direct deconvolution and sequential regression were rejected in favour of SLI because of the greater amount of processing required by them, a reappraisal of these techniques is required.

Direct deconvolution using a FFT is not a straightforward process in the case of the identification of the matched filter coefficients. The relationship between the filter coefficients, the autocorrelation of the noise and the signal to be detected is equivalent to the operation of convolution only in the interval [0,T], whereas for direct deconvolution the interval $[-\infty,\infty]$ is assumed. As we are only interested in the interval [0,T] it is necessary to multiply the signal to be detected by a window function to avoid ringing. This leads to the possibility of producing a sub-optimal filter. Alternatively, Whalen [17] suggested extending the signal to be detected so that it covers the interval $[-\infty,\infty]$, but points out this may lead to a sub-optimal and non-realizable filter (a non-realizable filter requires future inputs; provided a delay is permitted such a filter may be implemented digitally). One advantage

direct deconvolution does have over SLI is that it produces a result in a known time, and this is of great importance in real-time systems. However, it cannot be recommended in favour to sequential regression, which also has this property.

Returning to the discussion of viable alternative strategies to SLI one may see that sequential regression offers the best chance of improving the tracking characteristics of the generalised signal detector. Indeed, SLI and sequential regression are very similar techniques as they both improve a current estimate by adding an error correction vector. Sequential regression, however, requires more processing so that SLI was chosen given that the F100-L has a limited processing capability. With the improved processing power of the M68000 this argument against sequential regression becomes less valid, particularly when one considers the fact that sequential regression provides an estimate of the filter coefficients in a known time (unlike SLI, where an unknown number of iterations are required to provide a useful estimate). It would be fair to say that had the M68000 been available at the inception of the work on the generalised signal detector that sequential regression would have been the chosen technique employed in the filter identification process.

A more radical approach to improving the tracking characteristics of the identification procedure is gained by examining the way in which the autocorrelation of the noise is calculated. In the prototype the incoming non-stationary signal was divided into blocks over which a stationarity assumption could be made. The autocorrelation was built up from scratch by forming the partial sums of products of samples with various delays until an adequate average for the autocorrelation was obtained. Consequentially, the identification procedure may not begin until a sufficient number of samples have been read.

Consider, now, the possibility of an autocorrelation formed as a <u>moving</u> average, in that the current estimate of the autocorrelation is updated slightly whenever a new sample is read. Thus at all times there exists a valid estimate of the autocorrelation of the noise, implying that the identification process may be run continuously. This is a much more desirable concept, but it does present some problems. Firstly, as the autocorrelation is constantly changing the matched filter will appear to be non-stationary over the identification period. This eliminates system identification techniques such as direct deconvolution and regression, but fortunately SLI can cope with such problems. Secondly, it has been pointed out in previous chapters that either

the autocorrelation or filter sub-program (which are of similar duration) will determine the maximum sample rate of the generalised signal detector. If the autocorrelation process is the deciding factor then, when the detector is running at its maximum sample rate, there will be no processing time for the identification process, which run on the same processor. These arguments appear to indicate a triple-processor system with one processor for each of the filter, autocorrelation and identification processes.

The calculation of a moving average autocorrelation presents a number of problems. Consider an autocorrelation formed from a stream of input samples $x_1, x_2, \ldots, x_n$. This is given by

$$R_{xx}(k)_N = \frac{1}{N-k} \sum_{i=1}^{N-k} x_i x_{i+k} \qquad 11.1$$

When the next sample, x_{n+1}, is input this estimate of the autocorrelation will be superceded by an estimate formed from the input samples $x_2, x_3, \ldots, x_{n+1}$. Thus we have

$$R_{xx}(k)_{N+1} = \frac{1}{N-k} \sum_{i=2}^{N-k-1} x_i x_{i+k} \qquad 11.2$$

To find how $R_{xx}(k)_N$ may be changed to provide $R_{xx}(k)_{N+1}$ we form the difference of equations 11.1 and 11.2 and manipulate it to give

$$R_{xx}(k)_{N+1} = R_{xx}(k)_N + \frac{1}{N-k} (x_{N-k+1} x_{N+1} - x_1 x_{1+k}) \qquad 11.3$$

The above result demonstrates a simple way of generating a new estimate of the autocorrelation from a previous one. However, it requires access to the latest N+1 input samples, where N is the minimum number of samples which will provide a good average for the autocorrelation. Furthermore, for each point in the autocorrelation two multiplications and a division are required every time a sample is input as opposed to just one multiplication required by the more straightforward method. It may be noted that the product $x_1 x_{1+k}$ has already been formed in the calculation of previous autocorrelation values, and

therefore it may be stored rather than holding the samples x_1 and x_{1+k} and reforming the product. Unfortunately, this would require a total of 32K words of store (for N = 512 and k = 64), or in other words the entire addressing range of the F100-L ! It is possible that these products could be stored on MBM, but this would require half of the capacity of a 1M bit unit and, because of the access times of the MBM, may prove to be slower than recalculating the product.

In the autocorrelation calculation in the generalised signal detector prototype the division operation was replaced by a shift by using N+k samples rather than N. Thus $R_{xx}(0)$ was obtained from samples $x_1, \ldots, x_N$; $R_{xx}(1)$ was obtained from samples $x_1, \ldots, x_{N+1}$ and so on until $R_{xx}(k)$ which was formed from $x_1, \ldots, x_{N+k}$. This may be written as

$$R_{xx}(k)_{N+k} = \frac{1}{N} \sum_{i=1}^{N+k} x_i x_{i+k} \qquad 11.4$$

where $R_{xx}(k)_{N+k}$ is the estimate of the autocorrelation after the $N+k^{th}$ sample. It follows that

$$R_{xx}(k)_{N+K+1} = R_{xx}(k)_{N+k} + 1/N(x_{N+1}x_{N+k+1} - x_1 x_{1+k}) \qquad 11.5$$

If N is a power of 2 then 1/N may be implemented by a shift instruction. In terms of performance this means that the moving average autocorrelation is worse than the more straightforward method by a multiplication, an addition and a shift; these instructions should take about 25μS on the F100-L with hardware multiply (or less than 10μS on a M68000). In the author's opinion this is a small price to pay for an improved capacity to track non-stationary signals.

11.5 THE ROLE OF NON-VOLATILE STORE

The measurement technique which forms the generalised signal detector may be implemented on a variety of computer systems, either as a real-time or non-real-time process. Indeed, the amount of processing power required for the implementation of this technique would indicate that it is more suited to larger, more powerful computer systems than those based on microprocessors (given present day technology). However, it must be stressed that the development of the generalised signal detector was part of a group project to

provide an advanced portable instrument and a suitable application. It follows that the use of the signal detector, which is described in Part Two of this thesis, is dependent upon the availability of the advanced portable measurement system of which a bulk non-volatile store (as described in Part One) is an integral part.

Before a measurement may proceed the instrument must be "charged-up" with a program defining its task. This will be usually performed at a base rather than at the site where the measurement is to take place. Thus the first role of the non-volatile store is to retain the instrument's program until the start of the measurement. This is true of all applications of the RSP. However, the subsequent use of the non-volatile store will depend upon the application. For example, in section 11.4 it was suggested that products formed during the calculation of a moving average autocorrelation could be held within the non-volatile store to avoid their recalculation at a later stage. Although this suggestion was rejected, it would not be difficult to envisage other applications where the bulk non-volatile store would play a central part in the performance of a particular task. Finally, most applications will require results obtained at the measurement site to be retained for further analysis at the RSP base, and it is this use of the non-volatile store which will be examined further.

In a large number of applications (continuously running filters, etc.) there will be 1 result sample for each input sample. At the time of writing the maximum size of a non-volatile store based on a MBM unit in a practical system is 4M bits, or 512k bytes. For an EEG measurement problem, where a sample rate of about 170Hz is required, such an application could run for 51 minutes, reducing to approximately 41 minutes if 10 bit samples are used. For the generalised signal detector, however, it is likely that the incoming samples would be stored only if it was likely that a signal was present. In the case of the detection of the epilepsy precursor this would mean the storing of epochs of a duration of between 1 and 5 seconds, implying that at least 615 features could be detected before the non-volatile store became exhausted. Indeed, with the present system which has a 92k bit MBM unit as its non-volatile store and retaining 1 second epochs a total of 67 features could be detected; this is more than adequate. Employing the same system as a continuously running filter the non-volatile store would be exhausted after just over 1 minute, and thus the system would, in such an application, be of only limited value.

The nature of signal detection problems frequently means that there are long periods over which there is no signal present, and thus no input samples are stored (this is particularly true in the case of the detection of the epileptic precursor). In Chapter 3 it was shown that such applications favour the use of MBM as the storage medium for, as accesses to the store become less frequent, the power consumed by the MBM tends to zero. Conversely, as the non-volatile storage is used more frequently the high power consumption of the MBM (up to 11.5W) makes its employment as the storage medium less desirable.

The above arguments indicate that the advanced portable measurement system with MBM non-volatile backing store is more suitable to applications which require the infrequent storing of results, such as with the generalised signal detector, than to other applications which are more demanding on the non-volatile store.

12. CONCLUSIONS

It is feasible, with technology available at the time of writing, to design and build a fairly powerful "black-box" portable instrument based on two closely coupled 16-bit microprocessors. The availability of the hardware, however, is but a part of an overall strategy to provide experimenters who have limited computing knowledge with a complete computer aided measurement system from the initial conceptual stage of an experiment through to the final analysis of the results.

The final essential piece of the RSP hardware jigsaw - the non-volatile bulk store - can be supplied by the use of magnetic bubble memory technology. To complete the total system picture the framework of the software (the high level definer) is at an advanced stage. The work presented in this thesis has provided some extremely useful elements of the HLD (the autocorrelation and convolution processes) which may be easily included with minor changes, for example in parameter transfer. Finally, the addition of the more dedicated process of sequential learning identification (which may have uses other than that mentioned in this theses) has enabled the development of an easy to use generalised signal detector.

Although the performance of the generalised signal detector when applied to the problem of the detection of the epileptic precursor was a little disappointing the fact that it worked adequately is encouraging. It must be stressed that the generality and ease of problem definition is as important as performance. It is to be expected that specific solutions to such problems will always be at least as good as the generalised signal detector. The difference is that in order to provide a specific solution the experimenter must first acquaint himself with computer systems and techniques. With the generalised signal detector as part of the HLD the experimeter need only know the essential details of the signal before being able to build an instrument which will fulfil his measurement requirements.

13. SUGGESTIONS FOR FURTHER WORK

13.1 REVIEW OF PRESENT SYSTEM

Previous chapters have outlined the development of a portable RSP and its application to a class of measurement problems characterised by the detection of a known signal in undefined, and possibly variable, noise. The system is by no means ideal; indeed it is severely limited in terms of performance and, consequently, in the range of possible applications. Significant improvements are possible, and some are suggested in the following sections. First, however, it would be wise to outline the present system and its shortfalls.

The RSP hardware consists of a dual processor F100-L system with a total of 8k words of RAM, a bus extension unit, 8-bit DAC/ADC and 92K bits of non-volatile (MBM) storage. It is of the right size and weight for it to be considered portable. A "human" interface consisting of a small monitor screen and keyboard is also available. It is within the hardware that the most significant improvements can be made, as software changes tend to be limited in their scope to specific applications. Furthermore, much of the hardware pre-dates 1976 and is, compared to present-day technology, rather crude. More mundane problems, though no less relevant, concern the physical side of the RSP prototype in which the MBM unit and monitor screen are external to the main unit and the standard "finger-type" edge connectors could prove to be a liability in mechanically hostile environments.

The generalised signal detector provided encouraging, but still inadequate, results. When applied to the problem of the detection of the epileptic precursor results comparable to spike detectors, and superior to those obtained from "white-noise" matched filters, were obtained but improvements are still possible. The ability of the technique of sequential learning identification (SLI) to track non-stationarities has been shown to be acceptable provided the non-stationarities are not too rapid in nature. However, the major problems of the generalised signal detector lie less in its applicability than in its implementation. It should be remembered that the prototype software runs on a single processor F100-L system in what might be termed "process-sequential" mode (i.e. the processes of autocorrelation, filter identification and signal processing were performed in sequence rather than in parallel, as proposed). Furthermore, the processing time required for these processes are such that real-time operation on the problem of the detection of the epileptic precursor is not possible (the maximum sample rate of 22Hz would have to be increased

over threefold to cope with the bandwidth of the EEG signal). A combination of changes in hardware and implementation will overcome many, if not all, these disadvantages.

13.2 SUGGESTED IMPROVEMENTS TO THE HARDWARE

In Chapters 9 and 11 it was shown that immediate improvements in performance could be gained if hardware multiplication and division operations were provided. It was shown, for example that sample rates in excess of 90Hz could be obtained for the generalised signal detector when applied to the detection of the epileptic precursor; this is an adequate sample rate for real-time work on an EEG. A more general point, however, is the overall gain in processing power which would be obtained by opting for a different type of microprocessor as the RSP CPU. There is no doubt that an increase in processing power is desirable - indeed, it is essential before many applications can be tackled - but from a practical viewpoint a valid case can be made for retaining the F100-L based system until such time as it proves to be severely limited in its applications.

The most basic argument in favour of retaining the F100-L based RSP stems from its significantly different architecture compared to the later 16-bit microprocessors. It follows that the extensive work on dual F100-L systems is of little or no relevance to other multi-processor systems (the problem of DMA as outlined in Chapter 11 is of significance here), and would have to be repeated for the replacement device. Secondly, the later 16-bit microprocessors have been conceived as elements of an overall system. While this is a blessing for the majority of microprocessor applications the hardware overhead of bus transceivers and arbiters, DMA controllers, dynamic RAM controllers, general I/O controllers, etc., may prove too much for a portable system. Furthermore, in Chapter 11 it was noted that an obvious development in the generalised signal detector would be its implementation on a triple processor system. It would be most imprudent to move to such a system without first having examined the problems of dual processor operation (it will be shown later that the triple processor generalised signal detector is little different, in terms of degradation of performance, to the dual processor system).

The above arguments do not, of course, suggest that the possible gains obtained by developing an RSP based on, say, the MC68000 should be ignored and that work should continue only on the F100-L system. On the contrary, it has frequently

been stressed that increased processing power is essential for the long term success of the RSP concept as a measurement tool. However, there are many problems where the dual F100-L based RSP can provide an adequate solution and at the same time provide users with invaluable experience of measurement problems in non-laboratory environments. It is proposed, therefore, that two parallel paths of research are followed. One is the development of a more powerful RSP system and the second is employ the present RSP system in on-laboratory environment measurement problems. The latter course of action will require some changes to be made to the present RSP prototype (these changes are, in the main, cosmetic) and as this will directly affect the operation of the generalised dignal detector further consideration of these changes is given below.

13.2.1. Changes To The Microprocessor System

The most obvious improvement to the F100-L system is the provision of hardware multiply/divide. This has been achieved by Papworth (35) by using the Ferranti F101 MDU which, by its method of operation, appears to give the F100-L extra multiply and divide instructions. The method of implementation is to place the 40 pin F101 MDU on the processor board in "pick-a-back" style. While this is sufficient for the F101 to function correctly, doubts must be cast on the ruggedness of this configuration. It should not prove difficult to "harden" this design by implementing it on a custom PCB, and it is recommended that this be done as soon as possible.

One of the peculiarities of the F100-L system (peculiar, that is, compared to most other microprocessor systems) is that the data and address bus is asynchronous. Consequently, improvements in performance can be gained by reducing the access times of the RAM (in synchronous systems reducing access times further than the clock width will not provide any improvement). In the RAM boards supplied by Ferranti the interfacing is provided by Interface Sets (IS). The propogation delays through the IS can be anything up to 1.6μS. It is not difficult to see that a gain in processing power of at least 1.4μS per instruction would be possible if the interfacing were performed by, say, a bipolar PROM with some latches. This is equivalent to an approximate 25% increase in available processing power. Unfortunately, the relative complexity of the F100-L bus (which prompted Ferranti to develop the IS in the first place) implies that the design of such an interface is not a minor problem. However combining

this with the development of a low power RAM PCB based on, for example, 16k bit CMOS memories, would fully justify the design time, particularly as it would mean a step towards the "ultimate" RSP, which could be battery operated if required.

The propagation delays through the IS present another problem in the bus extension mode, which allows one processor to access memory on another's bus. In this case there are two IS connected "back-to-back", giving a total propagation delay of about 3.2μS. Results obtained by Mainwaring - Samwell indicate that this is the major cause of degradation in dual processor systems; consequently a reduction in this propagation delay could signficantly improve the performance of close-coupled multi-processor systems. Initial work on replacing the IS in the bus extension unit performed by final year students indicate a reduction of the propagation delay of about 3μS (36).

The recommendations given above are enhancements to the system rather than necessary changes which must be made before the RSP can be used in the way it was originally envisaged. However, one interesting point is that the suggestions cover system elements bought in from Ferranti. Should these changes be made, it would not be necessary to adhere to the physical structure of the F100-L system as dictated by Ferranti standards, for example PCB sizes. In particular, the type of edge connector used is not well suited to portable systems and should be replaced by more rugged pin connectors (it should be admitted, perhaps, that the author's views on this subject are greatly influenced by subsequent experience in industry, where "finger-type" edge connectors have proved to be a serious problem even in systems whose movements are limited to between production area, warehouse and site).

13.2.2. The DAC/ADC

The prototype RSP has an 8-bit DAC/ADC with sample times determined either by an external clock or by a hardware pre-settable on-board clock. In terms of a working system, this is sufficient for present applications. However, it is not altogether desirable to perform analogue I/O on this basis.

The first point to make is that 8 bits are not sufficient to encode/decode analogue signals in many applications and that 10 bits (an error of 1 part in 1000) should be considered the minimum. Secondly the concept of the RSP as a software defined instrument dictates that certain parameters be controlled by software. In particular, sample times should be software defined as they will vary from application to application. Certainly, an external clock may not always be available and the on-board clock is both limited in its range and tedious to preset.

Analogue I/O technology has advanced to the point where it should be comparatively simple to implement 10 bit multi-channel ADC/DAC with a software programmable real-time clock on a single board. If space (and power consumption) permit digital I/O should also be provided.

The problems of analogue I/O are inextricably linked with those of the transducers. Problems of non-stationarities for example (see Chapter 5) determine at which point in a measurement conversion takes place. Transducer theory itself is an area long overdue for an in-depth analysis and, although it is outside the scope of this thesis, it is to be hoped that work will begin on this topic in the near future. it should be remembered that the signal provided to the RSP (or any other instrument) is frequently only an analogue of a physical or chemical process. The closeness of the analogue to the actual process is, therefore, a limiting factor in the measurement.

13.2.3. Non-volatile Storage

The MBM control and memory boards are, because their sizes are different to the Ferranti PCB's, not contained in the RSP case. This is obviously not acceptable if the RSP is to be truly portable. Fortunately, it is possible to incorporate the control and up to four memory elements on a single PCB with little modification to the design supplied in Appendix; I; the MBM would then be incorporated into the RSP case and a truly portable RSP could then be said to exist. A single MBM memory element could be used without any changes being made to the given design and it might be wise to adopt this strategy in the immediate future. Apart from the speed of implementation involved by not making changes to a working design there is a further problem of PCB layout. As the MBM produces signals of only a few millivolts the problems of noise reduction are critical. Restricting

the design to just a single memory element should mean that the PCB layout is less critical than with four such devices. Nonetheless, it would still be wise to verify proposed PCB layouts with the manufacturers.

If more non-volatile storage is required then it may be considered desirable to change the MBM unit to, say, a 1M bit device. Certainly, this is recommended for the more powerful RSP. For present use, however, 92K bits is probably adequate. The prototype generalised signal detector, for example, is less than 2K words (32K bits) long, and it was shown in Chapter 11 that up to 67 spike-and-wave features could be stored in 92K bits. As the idea behind most of the suggestions contained here relate to providing a suitably portable RSP in as short a timescale as possible it is recommended that, for the time being, the designs given for the MBM unit are simply "hardened" onto PCB with no design changes.

One further point relating to the MBM unit should be made. As it stands, power is continually supplied to the MBM unit even when accesses are not being made. Although the MBM itself does not consume any power unless a transfer is in progress, the support devices do. As a consequence as the active duty cycle of the MBM tends to zero the power consumption, instead of tending to zero as suggested, will in fact tend to average at a value corresponding to the power consumed by the support devices multiplied by the power-up duty cycle of the RSP system. In future systems, where one would eventually hope to see battery operated RSP's, this shortcoming could be overcome in a fairly simple manner.

13.2.4. Other Hardware Considerations

The RSP as described in Part One of this thesis, is a total minicomputer/microprocessor/software system. One crucial element of the RSP structure which had not, at the time of writing, been developed is the minicomputer/microprocessor link. The problem has mainly been one of definition; it has proved difficult to establish what is required of the link. Fortunately, the development of the MBM unit has meant that programs and data may be transferred between a host computer (albeit a F100-L DOS Development system, which is somewhat less powerful than one would like) and the RSP. This means that a rigorous definition of the RSP link can be postponed until such time as more information is at hand with

regard to operational problems of the RSP, particularly with respect to establishing whether there is a need for real-time monitoring of the RSP by the host system.

Reference has also been made to a possible three processor system. As this possibility arises out of a consideration of the operation of the generalised signal detector, it will be covered later, even though it is a hardware problem. At this point, it is sufficient to say that for the given application there is a little difference in degradation between a triple and dual processor system.

13.3 SUGGESTIONS RELATING TO THE GENERALISED SIGNAL DETECTOR

The generalised signal detector implemented on the F100-L DOS Development system is purely a prototype used to examine the efficacy and performance of the technique. Obviously, any suggestions for further work must point in the direction of a usable real-time instrument. Many of the recommendations made in Section 13.2.1. will facilitate this aim, in that the required sample rates can be achieved by improvements in hardware. However, changes in the method of implementation are still essential.

Chapter 11 outlined a moving autocorrelation technique for keeping a constantly updated model of the background noise. This is necessary for the generalised signal detector to be able to handle signal detection problems in non-stationary noise (where the noise is stationary, the prototype software could be used with minimal changes). It was noted that this change would need the inclusion of a third processor, which would appear to be a major undertaking. Without wishing to trivialise the problems associated with such a major system change it should be pointed out that, in the case of a three processor generalised signal detector, the problems of the inter-processor contention are substantially the same as for the dual processor. To see why this is so, consider thre three processor system shown in figure 13.1 where each of the three processes in the generalised signal detector are performed on separate processors. An enhancement not provided on the dual processor prototype is partitioned common data areas (although this possibility has been considered by Mainwaring-Samwell in reference 34). Each partition contains one, and only one, of the three main data areas; i.e. the I/O buffer, the autocorrelation and the filter coefficients. It can be seen that each data area is accessed by only two of the processors, and, similarly, each processor access only two of the data areas. It follows that when a processor attempts to access a data

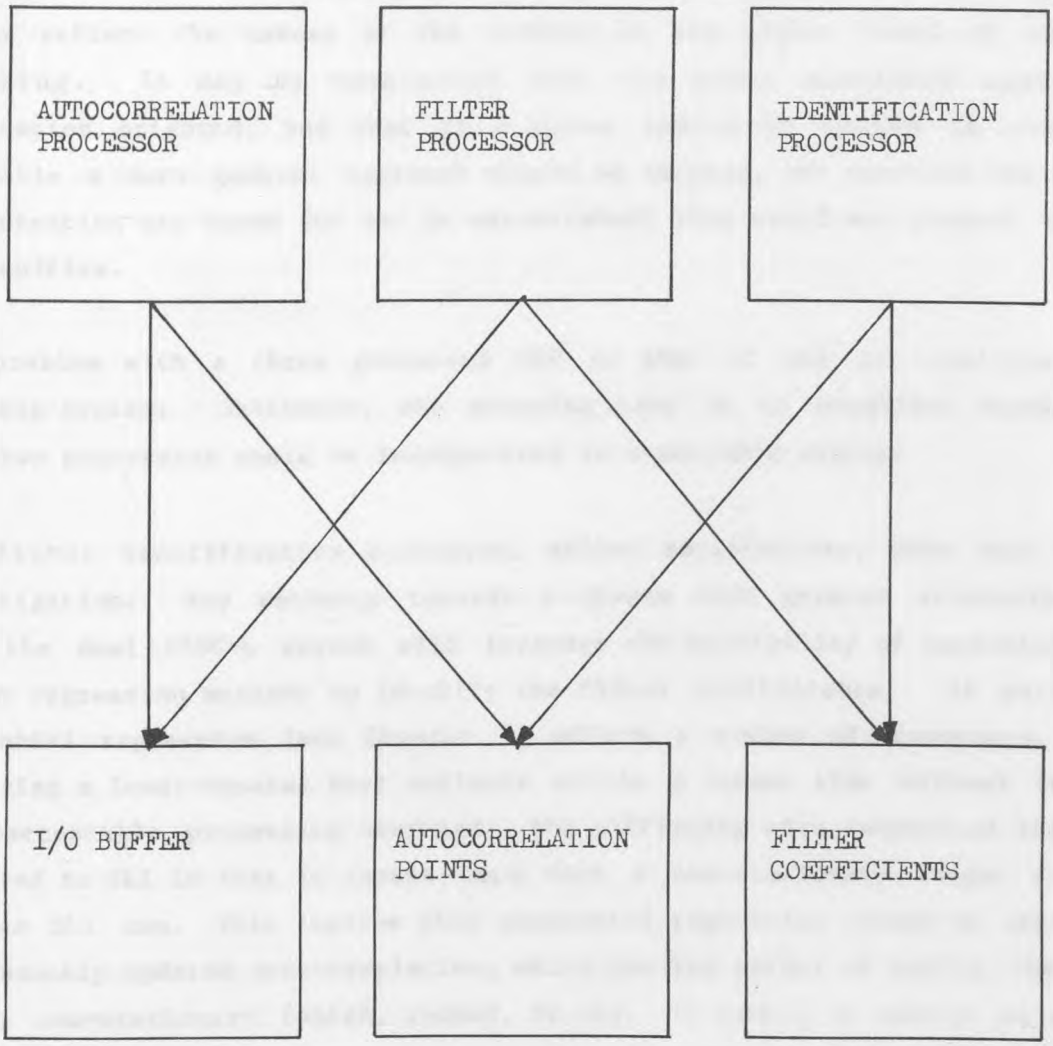


FIGURE 13.1 INTER-PROCESSOR CONTENTION IN TRIPLE-PROCESSOR GENERALISED SIGNAL DETECTOR

area it is in competition with only one other processor - precisely the same state of affairs one finds in the dual processor. This is, of course, a gross simplification and overlooks problems such as bus contention. However, it does fairly reflect the nature of the problem at the higher level of semaphore handling. It may be considered that the above mentioned approach is application oriented, and that if a three processor system is considered desirable a more general approach should be adopted, but provided the problems of contention are known (or can be established) this would not present too many difficulties.

One problem with a three processor RSP is that it may not lend itself to a portable system. Obviously, one pressing task is to establish whether more than two processors could be incorporated in a portable system.

The filter identification technique, whilst satisfactory, does bear further investigation. Any tendency towards a system with greater processing power than the dual F100-L system will increase the possibility of employing either FFT or regression methods to identify the filter coefficients. In particular, sequential regression (see Chapter 7) offers a number of advantages, such as producing a least-squares best estimate within a known time without incurring an unacceptable processing overhead. The difficulty with sequential regression compared to SLl is that it cannot cope with a non-stationary target function, whereas SLI can. This implies that sequential regression cannot be used with a continuously updated autocorrelation, which has the effect of making the filter appear non-stationary (which, indeed, it is). It would, of course, be possible to "freeze" the moving autocorrelation at the start of the identification process and use this estimate to obtain the filter coefficients. This method of filter identification would, therefore, forsake the advantages of the moving autocorrelation as outlined in Chapter 11 but would gain a greater degree of predictability within the identification process. Only by examining the results of actual experiments will it be possible to see whether this is an important consideration.

REFERENCES

1. BRIGNELL AND RHODES; "LABORATORY ON-LINE COMPUTING"; INTERTEXT, 1975

2. BRIGNELL; "THE DIGITAL COMPUTER IN THE LABORATORY"; QUEST NO. 6; SEPTEMBER 1968

3. BRIGNELL, COMLEY AND YOUNG; "THE ROVING SLAVE PROCESSOR"; MICROPROCESSORS, VOL 1; DECEMBER 1976

4. COMLEY; "PORTABLE COMPUTERS FOR REAL-TIME SIGNAL PROCESSING: EEG ANALYSIS AS A CASE STUDY"; Ph.D. THESIS, THE CITY UNIVERSITY, 1978

5. YOUNG; "ROVING SLAVE PROCESSORS FOR COMPUTER AIDED MEASUREMENT: PRINCIPLES AND DESIGN CONSIDERATIONS"; Ph.D. THESIS, THE CITY UNIVERSITY, 1979

6. FERRANTI LTD; HSM 150, F100-L HARDWARE AND SYSTEM MANUAL

7. JULIUSSON, LEE AND COX; "BUBBLES APPEARING FIRST AS MICROPROCESSOR MASS STORAGE"; ELECTRONICS, AUGUST 4, 1977

8. IBM RESEARCH RELEASE; "1K BIT CHIP DEMONSTRATES STORAGE FUNCTIONS OF BUBBLE LATTICE"; NOVEMBER 17, 1977

9. IBM RESEARCH RELEASE; "IBM SCIENTISTS FORM VERY SMALL, 0.4 MICRON, MAGNETIC BUBBLES"; MAY 9, 1978

10. CAPECE; "BIG BUBBLE BOWS"; ELECTRONICS, SEPTEMBER 14, 1978

11. BRYSON, CLOVER AND LEE; "MEGABIT BUBBLE MEMORY GET SUPPORT FROM LSI FAMILY", ELECTRONICS, APRIL 26, 1979.

12. LIU; "EFFECT OF FINITE WORDLENGTH ON THE ACCURACY OF DIGITAL FILTERS A REVIEW"; DIGITAL SIGNAL PROCESSING, RABINER AND RADER (EDITORS) IEEE PRESS, 1972

13. KNIGHT; UNDERGRADUATE PROJECT REPORT; THE CITY UNIVERSITY, 1977

14. CHEESMAN;UNDERGRADUATE PROJECT REPORT; THE CITY UNIVERSITY, 1978

15. FERRANTI LTD; MDU DATA SHEET

16. PAPOULIS; "PROBABILITY, RANDOM VARIABLES AND STOCHASTIC PROCESSES"; McGRAW-HILL, 1965

17. WHALEN; "DETECTION OF SIGNALS IN NOISE"; ACADEMIC PRESS, 1971

18. BARLOW; "COMPUTERIZED CLINICAL ELECTROENCEPHALOGRAPHY IN PERSPECTIVE"; IEEE TRANSACTIONS ON BIOMEDICAL ENGINEERING, VOL. BME-26, NO. 7, pp 377-391, JULY 1979

19. VERA AND BLUME; "A CLINICALLY EFFECTIVE SPIKE RECOGNITION PROGRAM : ITS USE AT ELECTROCORTICOGRAPHY", ELECTROENCEPHALOGR. CLIN. NEUROPHYSIOL, VOL. 45, pp 545-548, 1978

20. KAWABATA;"A NONSTATIONARY ANALYSIS OF THE ELECTROENCEPHALOGRAM", IEEE TRANS. BIOMEDICAL ENGINEERING, VOL. BME-20, No. 6, pp 444-452, NOV. 1973

21. ELUL; "GAUSSIAN BEHAVIOUR OF THE ELECTROENCEPHALOGRAM: CHANGES DURING PERFORMANCE OF MENTAL TASK"; SCIENCE, VOL. 164, pp. 328-331, APR-JUNE 1969

22. HUBER, KLEINER, GASSER AND DUMERMUTH; "STATISTICAL METHODS FOR INVESTIGATING PHASE RELATIONS IN STATIONARY STOCHASTIC PROCESSES"; IEEE TRANS., AUDIO ELECTROACOUT., VOL. AU-19, pp 78-96, MARCH 1971

23. CHATRIAN, BERGAMINI, DONDEY, KLASS, LENNOX-BUCHTAL AND PETERSEN; "A GLOSSARY OF TERMS MOST COMMONLY USED BY CLINICAL ELECTROENCEPHALOGRAPHERS"; ELECTROENCEPHALOGR. CLIN. NEUROPHYSIOL., VOL. 37, pp. 538-548, 1974

24. GOTMAN, GLOOR AND SCHAUL; "COMPARISON OF TRADITIONAL READING OF THE EEG AND AUTOMATIC RECOGNITION OF EPILEPSY ACTIVITY"; ELECTROENCEPHALOGR. CLIN. NEUROPHYSIOL., VOL. 44, pp. 48-60,1978

25. QUILTER, MACGILLIURAY AND WADBROOK; "THE REMOVAL OF EYE MOVEMENT ARTEFACT FROM EEG SIGNALS USING CORRELATION TECHNIQUES"; IEE CONFERENCE PUBLICATION No. 159, COLLOQUIUM ON RANDOM SIGNAL ANALYSIS, pp 93-100, 1977

26. CARRIE; "A HYBRID COMPUTER SYSTEM FOR DETECTING AND QUANTIFYING SPIKE AND WAVE EEG PATTERNS"; ELECTROENCEPHALOGR. CLIN. NEUROPHYSIOL., VOL. 33, pp. 339-341, 1972

27. BARLOW; "EEG TRANSIENT DETECTION BY MATCHED INVERSE DIGITAL FILTERING"; ELECTROENCEPHALOGR. CLIN. NEUROPHYSIOL., VOL. 48, pp. 246-248, 1980

28. POWELL; "A SURVEY OF NUMERICAL METHODS FOR UNCONTRAINED OPTIMIZATION"; PERSPECTIVES ON OPTIMIZATION, A.M. GEOFFRION (ED.), ADDISON-WESLEY, 1972

29. FLETCHER; "A REVIEW OF METHODS FOR UNCONSTRAINED OPTIMIZATION"; OPTIMIZATION, R. FLETCHER (ED.), ACADEMIC PRESS, 1969

30. PUN; "INTRODUCTION TO OPTIMIZATION PRACTICE"; WILEY AND SONS, 1969

31. NASH; "COMPACT NUMERICAL METHODS FOR COMPUTERS"; ADAM HILGER LTD., 1979

32. TOCHER; "THE ART OF SIMULATION"; HODDER AND STOUGHTON, 1963

33. GRAUPE; "THE IDENTIFICATION OF SYSTEMS"; VAN NOSTRAND RETNHOLD, 1972

34. MAINWARING-SAMWELL; "ASPECTS OF CONTENTION IN CLOSELY COUPLED PROCESSOR SYSTEMS"; Ph.D. THESIS; THE CITY UNIVERSITY, 1980

35. PAPWORTH; Ph.D. THESIS (UNPUBLISHED)

36. COMLEY; PRIVATE COMMUNICATION, 1981

ACKNOWLEDGEMENTS

The author would like to express his gratitude to the Department of Electrical and Electronic Engineering, The City University for the use of its facilities, and to British Rail for sponsoring the majority of the work presented in this thesis.

The author would like to thank all the members of the Computer Aided Measurement research group who, under the guidance of Professor Brignell (now of Southampton University), provided a most stimulating environment. I do not think that any members of the CAM research group would object to a special mention for Dr Richard Comley whose energy, enthusiasm and willingness to assist enabled us to achieve more than we might otherwise (often, it must be said, at the expense of his own research interests).

The author would also like to thank his current employer, Plessey Communication Systems Ltd., and in particular Dr Peter Bailes (now of Rolls-Royce) for permission to employ company staff and resources to produce this thesis whenever possible.

Finally, thanks go to the author's family and friends for their unswerving support and encouragement over an inordinately long period, particularly to the author's wife, Sharon, whose encouragement ensured completion of this thesis only to discover that she would have to type it!

APPENDIX I: MBM INTERFACE HARDWARE

APPENDIX I: MBM INTERFACE HARDWARE

I.1 PURPOSE OF THE CIRCUIT

Two PCB's are supplied by the manufacturer for magnetic bubble memory (MBM) evaluation purposes. One of these (BKA0203A) is the memory unit itself. It contains the MBM, coil drivers, sense amplifier and associated circuitry (e.g. diode arrays, R-C networks, etc.). The second PCB (BCA0200A) is the memory controller which contains all the logic required to interface a BKA0203A on to a TMS 990 series system. The major component is the TMS 5502 (also referred to as the TMS 9916) MBM controller integrated circuit.

Two possible solutions to the problem of providing an MBM interface to the F100-L were evident:

1) provide a mapping from the F100-L bus to a TMS 990 bus so that use could be made of the BCA0200A
2) provide a new controller PCB by using just the TMS 5502.

Although option (2) is more elegant, option (1) was chosen on the grounds of simplicity and because the 2 board evaluation kit was more readily available than the TMS 5502 it contained.

The F100-L bus differs significantly from that of the TMS 990. The F100-L bus is asynchronous and multiplexes the address and data whereas the TMS 990 has separate, synchronous address and data busses. However, it is not a difficult task to provide a mapping when an Interface Set (IS) is used as this latches address and data information off the F100-L bus and presents them separately.

By using the IS some of the logic on the BCA0200A becomes redundant. First, the address recognition logic is no longer necessary, so the PROM's which perform this function were removed. Secondly, as there is no need to synchronise with the system bus this logic could also be removed. The changes to the BCA0200A are explained in Section I.2.

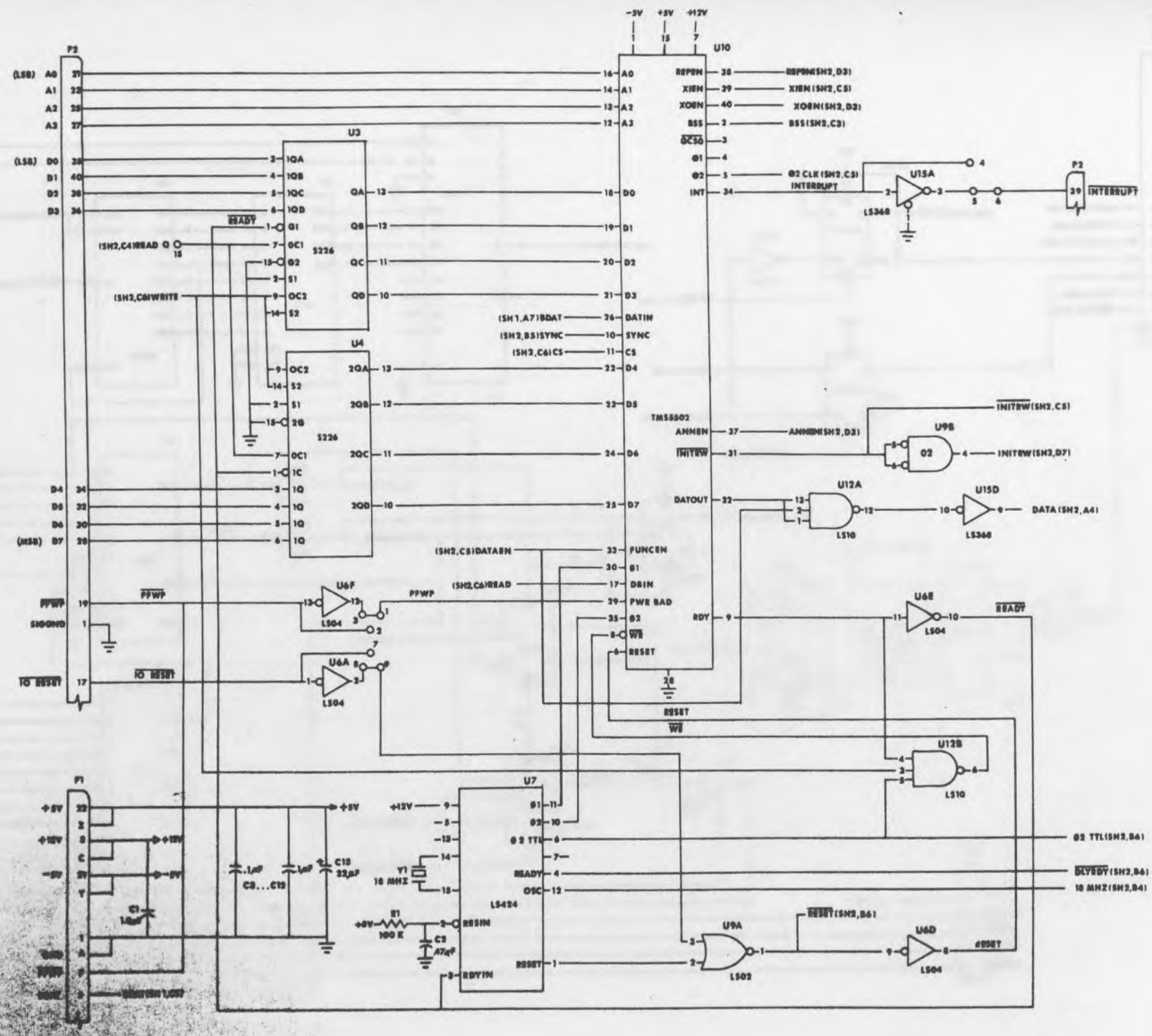
I.2 CHANGES TO THE BCA0200A CONTROLLER

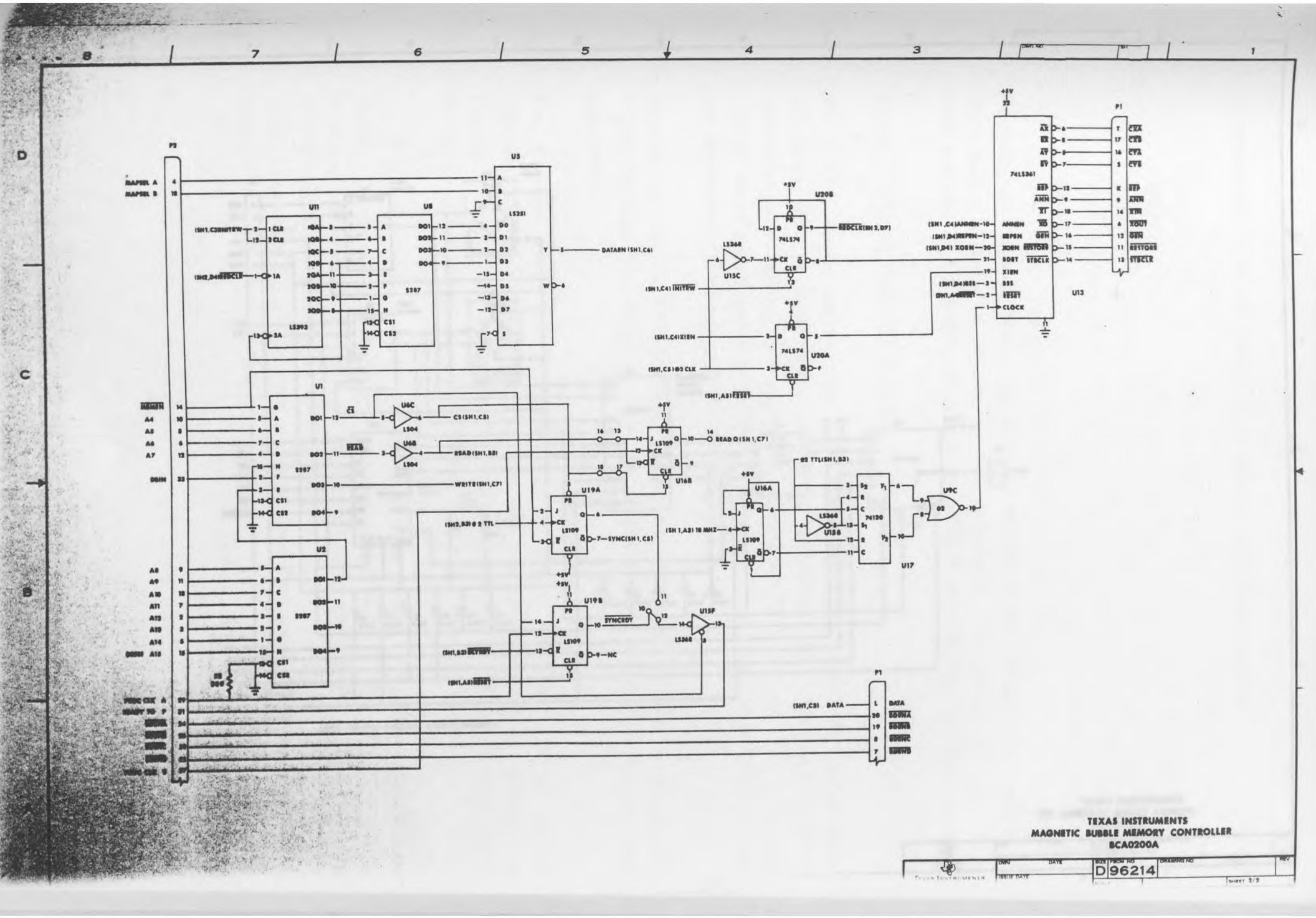
The address recognition logic uses 2 permanently selected bipolar PROM's to generate 3 output signals from the 12 most significant address lines. The three output signals are $\overline{CS}$, $\overline{READ}$ and WRITE. As the IS performs the address recognition and provides the necessary handshaking, read and write signals, it is a simple task to generate the 3 required signals directly. The need for the bipolar PROM's was thereby obviated.

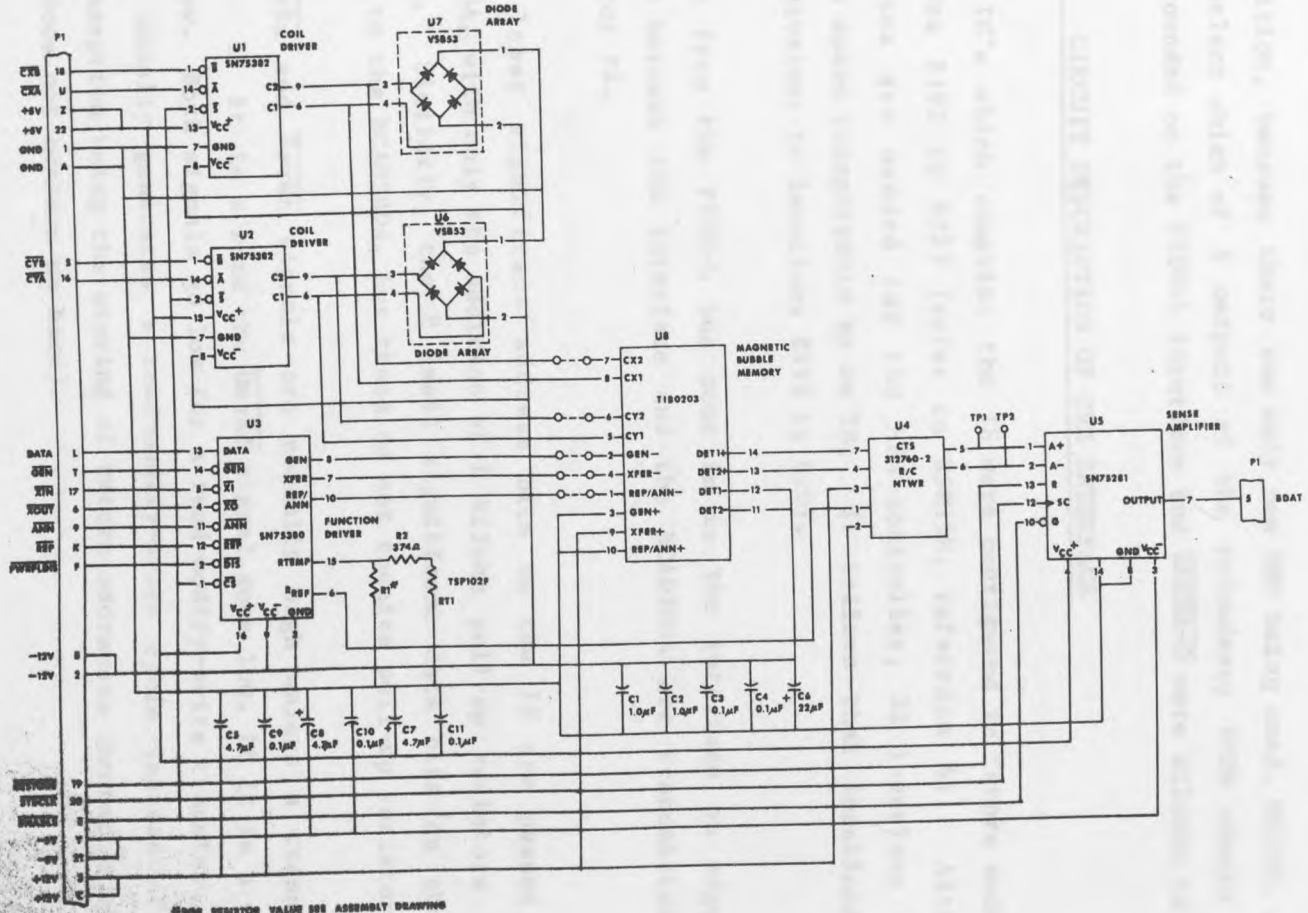
The problem of transmitting $\overline{CS}$, $\overline{READ}$ and WRITE fron the F100-L interface to the correct place on the controller PCB was overcome by using pins 12 and 14 on the edge connector P2 which, as they were previously used as part of the address, were now available. It should be noted that $\overline{READ}$ and WRITE are in fact identical; one wonders why the manufacturer believed that two outputs from the PROM's were required. Obviously, it was necessary to transmit only 1 signal, referred to as $\overline{READ}$/WRITE, between the interface and the controller. From sheet 2 of the Texas Instruments drawing 96214 for the BCA0200A it can be seen that the only modifications required were to connect on IC U1

a) pin 1 to pin 12
b) pin 4 to pins 10 and 11

The only other modification required is to remove the controller/bus synchronisation. This is catered for in the design of the BCA0200A. Again referring to sheet 2 of drawing 96214 (grid reference B5) the link shown between points 10 and 12 was broken and a link between points 11 and 12 made. Similarly, the link shown between points 14 (sheet 2) and 15 (sheet 1) was broken and a link between points 16 (sheet 1) and 15 made, thereby making IC U16B redundant. With these changes made there was no need to provide connection to PROC CLK A, PROC CLK B or READY TO P from the interface. The controller indicates that it is ready for a transfer by making $\overline{READY}$ go "low" and this was used instead of READY TO P. This was transmitted to the F100-L interface by using pin 8 of P2. Hence pin 10 of U6 was connected to pin 6 of U1.


P2
U3
U4
U10
U15A
U9B
U12A
U15D
U6F
U6A
U6E
U12B
U9A
U6D
U7
P1
TEXAS INSTRUMENTS
MAGNETIC BUBBLE MEMORY CONTROLLER
BCA0200A
D 96214


TEXAS INSTRUMENTS
MAGNETIC BUBBLE MEMORY CONTROLLER
BCA0200A
D96214



*FOR RESISTOR VALUE SEE ASSEMBLY DRAWING

TEXAS INSTRUMENTS
92K MAGNETIC BUBBLE MEMORY
BKA0203A

TEXAS INSTRUMENTS	DWN	DATE	SIZE / FSCM NO. D 96214	DRAWING NO.
	TRBLE DATE		SCALE	SHEET 1/1

To summarise, it is not necessary to use the following signals:

A4-15
$\overline{\text{MEMEN}}$
DBIN
PROC CLK A-B
READY TO P

In addition, because there was only one MBM being used, MAPSEL A and MAPSEL B, which select which of 4 outputs of the redundancy PROM should be used, were both grounded on the F100-L interface and $\overline{\text{BDENA-D}}$ were allowed to float to 5V.

I.3 CIRCUIT DESCRIPTION OF THE INTERFACE

The 3 IC's which comprise the IS were configured in "store mode" to recognise addresses 8192 to 8223 (refer to HSM150, reference 6). Although only 16 locations are needed for the MBM controller, 32 locations is the minimum address space recognisable by an IS. It follows that locations 8208 to 8223 are equivalent to locations 8192 to 8207.

Signals from the F100-L bus come in to the interface on edge connector P1. Signals between the interface and the BCA0200A are transmitted via the edge connector P2.

The 4 least significant address bits on the IS are passed direct to the BCA0200A, with only the addition of 1 kilohm pull-up resistors as required by the IS. Similarly, the 8 least significant data bits on the IS are passed direct to the BCA0200A, but these do not require pull-up resistors.

The $\overline{\text{DevRd}}$ and $\overline{\text{DevWt}}$ signals are normally high unless a transfer is to take place. If it is a read the $\overline{\text{DevRd}}$ signal goes low. If it is a write then $\overline{\text{DevWt}}$ goes low. Both signals go low for a read-modify-write transfer. In fact, the F100-L usually generates a read-modify-write cycle instead of a write cycle (the exception being the storing of return addresses during a subroutine call, which does not concern us here).

The handshake signals $\overline{\text{JDev}}$ (from the IS) and $\overline{\text{KDev}}$ (from the interface) are used as follows. During a read cycle the sequence is:

i) $\overline{\text{JDev}}$ goes low: transfer request
ii) $\overline{\text{KDev}}$ goes low: data on bus
iii) $\overline{\text{JDev}}$ goes high: data accepted
iv) $\overline{\text{KDev}}$ goes high: transfer complete

For a write cycle (ignoring the read part of the read-modify-write) the sequence is:

i) $\overline{\text{JDev}}$ goes low: transfer request
ii) $\overline{\text{KDev}}$ goes low: ready to accept
iii) $\overline{\text{JDev}}$ goes high: data on bus
iv) $\overline{\text{KDev}}$ goes high: data accepted, transfer complete

I.3.1 GENERATING $\overline{\text{KDev}}$

From the signal descriptions given in section I.3 it can be seen that $\overline{\text{KDev}}$ should go low when

a) Write: $\overline{\text{DevWt}}$ is low and $\overline{\text{JDev}}$ goes low
b) Read: $\overline{\text{DevWt}}$ is high, $\overline{\text{DevRd}}$ is low, $\overline{\text{JDev}}$ is low and $\overline{\text{READY}}$ goes low

Similarly, $\overline{\text{KDev}}$ should go high when:

a) Write: $\overline{\text{DevWt}}$ is low, $\overline{\text{JDev}}$ is high and $\overline{\text{READY}}$ goes low
b) Read: $\overline{\text{DevRd}}$ is low, $\overline{\text{DevWt}}$ is high and $\overline{\text{JDev}}$ goes high

This is shown on "F100-L/BCA0200A I/F" drawings, sheets 1 and 2.

I.3.2 GENERATING $\overline{\text{MEMEN/CS}}$

$\overline{\text{MEMEN/CS}}$ is an "active low" signal which replaces MEMEN on a TMS 990 bus with a chip select function. In simple terms it is held low while a transfer of either type takes place. For a write cycle it must be held low until the controller indicates, via $\overline{\text{READY}}$, that the data has been

accepted. Conversely, for a read cycle it must be held low until the processor has accepted the data. Hence $\overline{\text{MEMEN/CS}}$ will be low when:

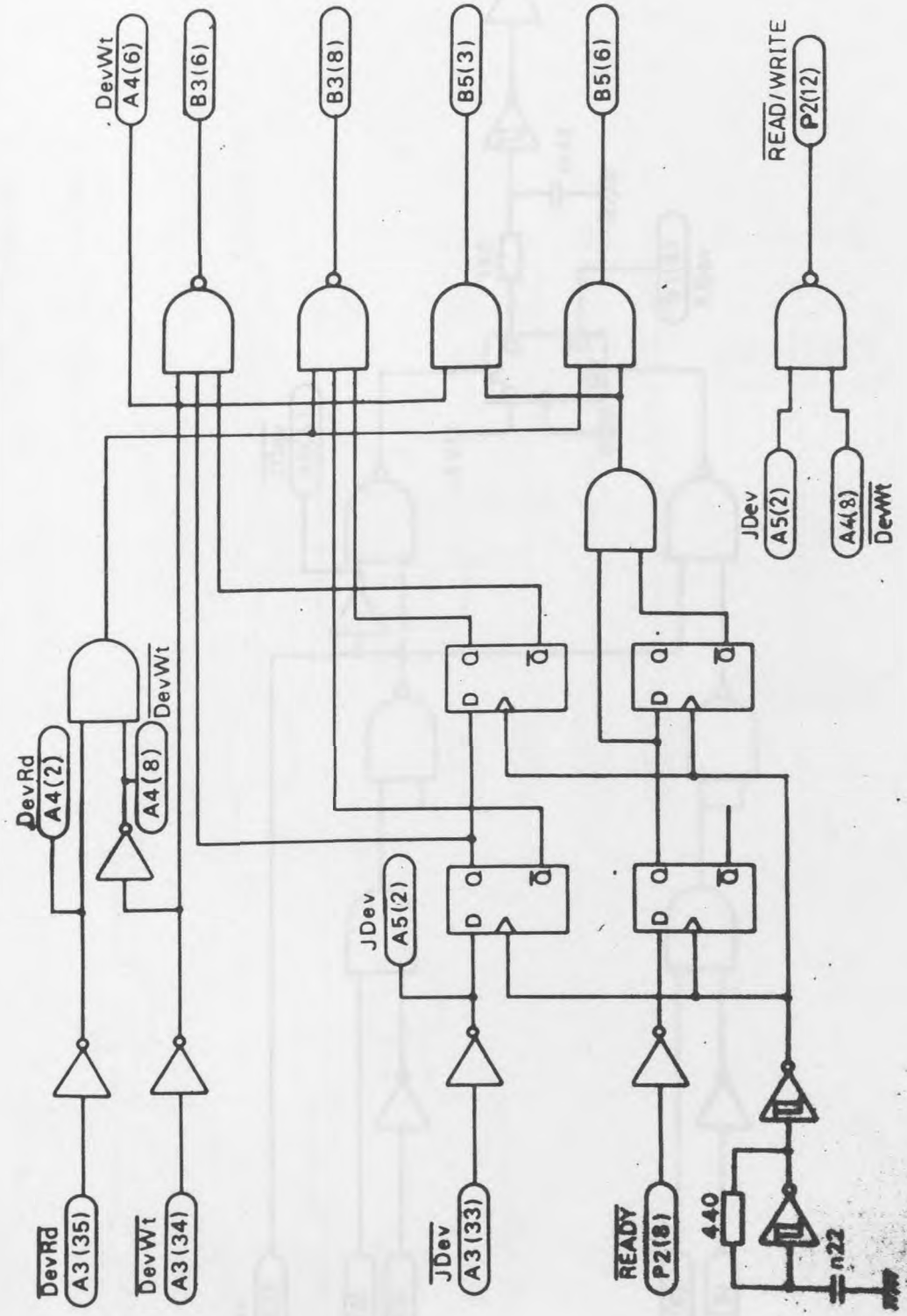
a) Write: $\overline{\text{DevWt}}$ is low, $\overline{\text{KDev}}$ is low and $\overline{\text{JDev}}$ is high

b) Read: $\overline{\text{DevWt}}$ is high, $\overline{\text{DevRd}}$ is low and $\overline{\text{JDev}}$ is low

This is shown on sheet 3 of "F100-L/BCA0200A I/F" drawings.

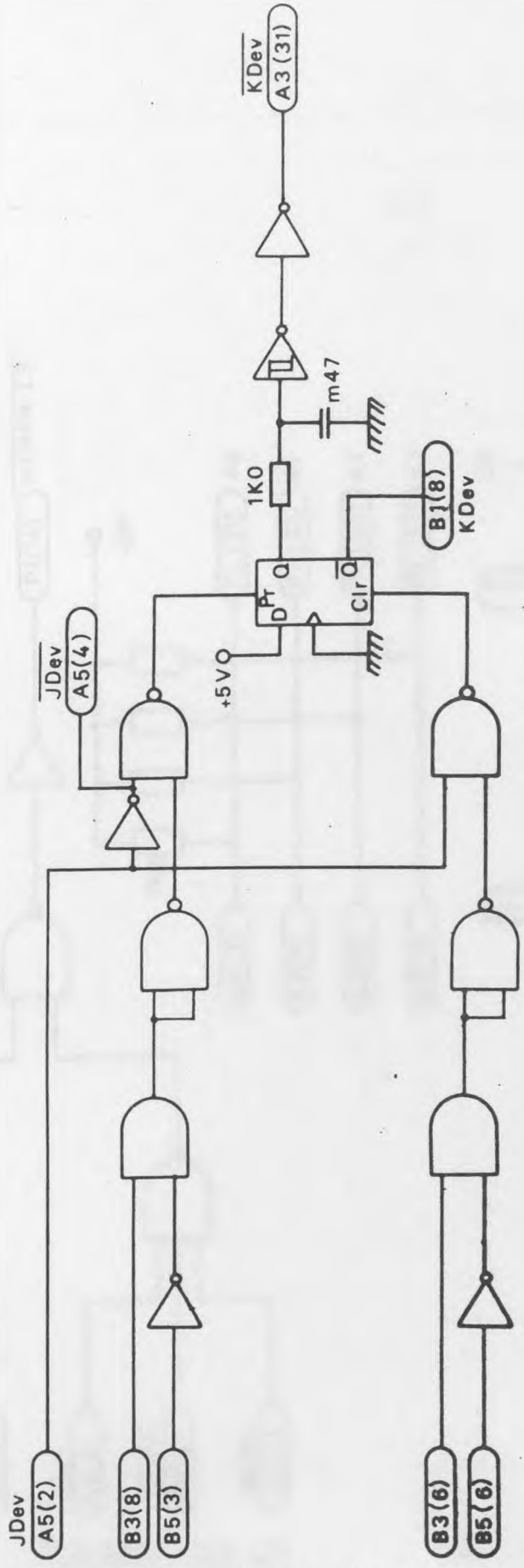
I.4 F100-L/BCA0200A I/F BOARD WIRING

Sheet 4 of the "F100-L/BCA0200A I/F" drawings shows the board layout (not to scale). The interconnections are detailed in the "IC Interconnection Charts". Connections are shown only once in these charts. For example a connection between pin 20 on IC A1 to pin 17 on IC A3 is shown as a connection from A1 to A3, but there is no entry for the connection from A3 to A1.

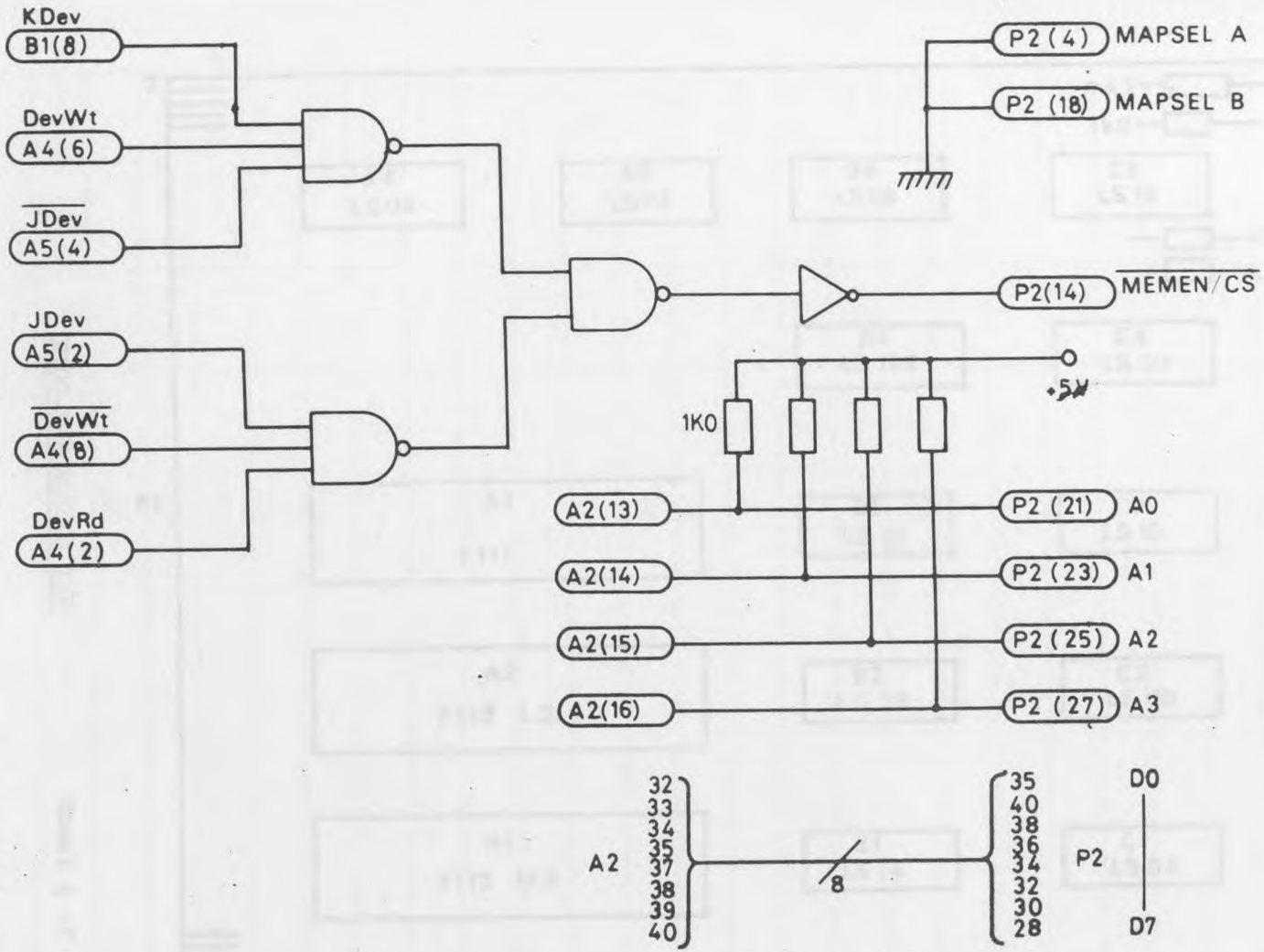


F100-L/BCA0200A I/F

Sheet 1 of 4



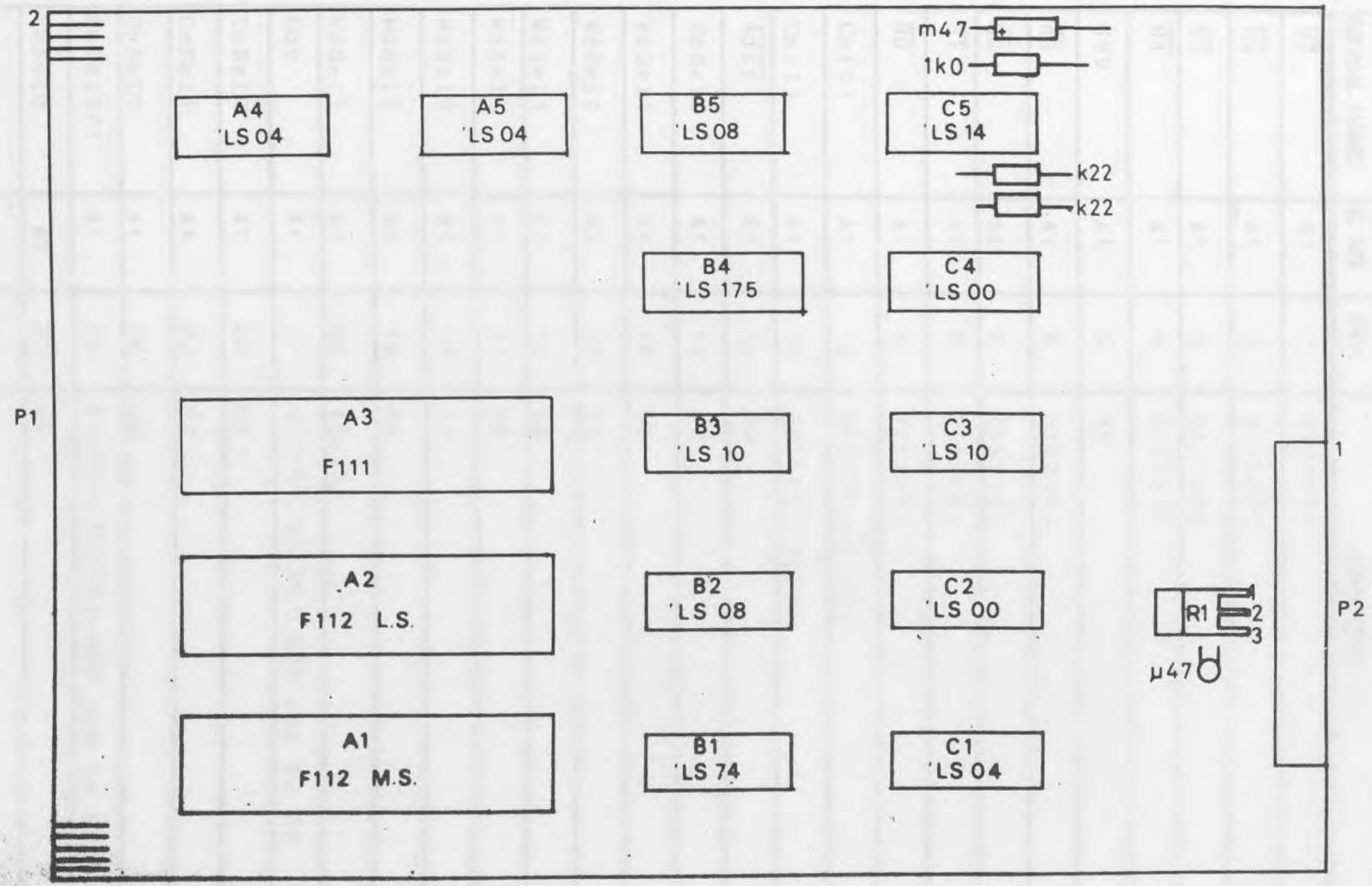
F100-L/BCA0200A I/F



F100-L/BCA0200A I/F

Sheet 3 of 4

F100-L/BCA0200A I/F



SIGNAL NAME	IC NO.	PIN	CONNECTED TO
$\overline{H7}$	A1	1	P1(118)
$\overline{H6}$	A1	2	P1(117)
$\overline{H5}$	A1	3	P1(116)
$\overline{H4}$	A1	4	P1(115)
GND	A1	5	0V
$\overline{H3}$	A1	6	P1(114)
$\overline{H2}$	A1	7	P1(113)
$\overline{H1}$	A1	8	P1(112)
$\overline{H0}$	A1	9	P1(111)
Cn(0)	A1	10	A2(10),A3(10)
Cn(1)	A1	11	A2(11),A3(11)
$\overline{AdCy}$	A1	12	NC
WdSel0	A1	13	5V
WdSel1	A1	14	5V
WdSel2	A1	15	5V
WdSel3	A1	16	5V
WdSel4	A1	17	5V
WdSel5	A1	18	5V
WdSel6	A1	19	5V
WdSel7	A1	20	A3(17)
Eqv	A1	21	A2(21),A3(24),680 ohm to 0V
CnSel0	A1	22	0V
CnSel1	A1	23	0V
CnSel2	A1	24	0V
FnSel0/1	A1	25	A1(26),A2(26),680 ohm to 5V
FnSel2	A1	27	0V
	A1	28	NC
EqvSb	A1	29	A2(29),A3(16)

IC INTERCONNECTION CHART

SIGNAL NAME	IC NO.	PIN	CONNECTED TO
Vcc	A1	30	5V
GND	A1	31	0V
	A1	32	NC
	A1	33	NC
	A1	34	NC
	A1	35	NC
GND	A1	36	0V
	A1	37	NC
	A1	38	NC
	A1	39	NC
	A1	40	NC
XXXXXXXXXXX	XXXXXX	XXXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
$\overline{H15}$	A2	1	P1(110)
$\overline{H14}$	A2	2	P1(109)
$\overline{H13}$	A2	3	P1(106)
$\overline{H12}$	A2	4	P1(105)
GND	A2	5	0V
$\overline{H11}$	A2	6	P1(104)
$\overline{H10}$	A2	7	P1(103)
$\overline{H9}$	A2	8	P1(102)
$\overline{H8}$	A2	9	P1(101)
	A2	12	NC
WdSel8	A2	13	P2(21),1 kohm to 5V
WdSel9	A2	14	P2(23),1 kohm to 5V
WdSel10	A2	15	P2(25),1 kohm to 5V
WdSel11	A2	16	P2(27),1 kohm to 5V
WdSel12	A2	17	NC
WdSel13	A2	18	0V

SIGNAL NAME	IC NO.	PIN.	CONNECTED TO
WdSel14	A2	19	0V
WdSel15	A2	20	0V
CnSel0	A2	22	0V
CnSel1	A2	23	0V
	A2	24	NC
FnSel0	A2	25	0V
FnSel2	A2	27	0V
	A2	28	NC
Vcc	A2	30	5V
GND	A2	31	0V
DevH8	A2	32	P2(35)
DevH9	A2	33	P2(40)
DevH10	A2	34	P2(38)
DevH11	A2	35	P2(36)
GND	A2	36	0V
DevH12	A2	37	P2(34)
DevH13	A2	38	P2(32)
DevH14	A2	39	P2(30)
DevH15	A2	40	P2(28)
XXXXXXXXXX	XXXXXX	XXXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXX
	A3	1	NC
	A3	2	NC
	A3	3	NC
	A3	4	NC
	A3	5	NC
	A3	6	NC
	A3	7	NC
$\overline{\text{DevRq}}$	A3	8	1 kohm to 5V

IC INTERCONNECTION CHART

SIGNAL NAME	IC NO.	PIN	CONNECTED TO
	A3	9	NC
	A3	12	NC
	A3	13	NC
	A3	14	NC
$\overline{\text{IRd}}$	A3	15	1 kohm to 5V
AdSb	A3	18	0V
	A3	19	NC
Vcc	A3	20	5V
	A3	21	NC
$\overline{\text{Rs}}$	A3	22	P1(100)
	A3	23	NC
K(Pas)	A3	25	P1(98)
$\overline{\text{J(Acv)}}$	A3	26	P1(97)
$\overline{\text{J(Pas)}}$	A3	27	P1(94)
$\overline{\text{K(Acv)}}$	A3	28	P1(93)
GND	A3	29	0V
$\overline{\text{WtExt}}$	A3	30	P1(92)
$\overline{\text{KDev}}$	A3	31	C1(12)
$\overline{\text{JDev}}$	A3	33	A5(1)
$\overline{\text{DevWt}}$	A3	34	A4(5)
$\overline{\text{DevRd}}$	A3	35	A4(1)
GND	A3	36	0V
	A3	37	NC
	A3	38	NC
	A3	39	NC
	A3	40	NC
XXXXXXXXXX	XXXXXX	XXXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

IC INTERCONNECTION CHART

SIGNAL NAME	IC NO.	PIN.	CONNECTED TO
	A4	2	C3(10), B2(1)
	A4	6	A4(9), B5(2), B3(3), C3(2)
	A4	7	0V
	A4	8	C4(1), C3(9), B2(2)
	A4	14	5V
XXXXXXXXXXXX	XXXXXX	XXXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXX
	A5	2	A5(3), B4(4), C4(2), C3(11), C2(9)
	A5	4	C4(9), C3(13)
	A5	5	P2(8)
	A5	6	B4(13)
	A5	7	0V
	A5	8	B2(4)
	A5	9	B5(3)
	A5	10	B2(9)
	A5	11	B5(6)
	A5	12	P2(14)
	A5	13	C4(6)
	A5	14	5V
XXXXXXXXXXXX	XXXXXX	XXXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXX
	B5	1	B5(4), B2(11)
	B5	5	B3(9), B2(3)
	B5	7	0V
	B5	14	5V
XXXXXXXXXXXX	XXXXXX	XXXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXX
	C5	3	220 ohm to pad;pad via 220 ohm
			to C5(4) and C5(5),n22F to 0V
	C5	6	B4(9)
	C5	7	0V

IC INTERCONNECTION CHART

SIGNAL NAME	IC NO.	PIN	CONNECTED TO
	C5	8	C1(13)
	C5	9	1kohm to B1(9),470µF to 0V
	C5	14	5V
XXXXXXXXXX	XXXXXX	XXXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXX
	B4	1	5V
	B4	2	B4(5),B3(4)
	B4	3	B3(11)
	B4	6	B3(5)
	B4	7	B3(10)
	B4	8	0V
	B4	11	B2(12)
	B4	12	B4(15),B2(13)
	B4	16	5V
XXXXXXXXXX	XXXXXX	XXXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXX
	C4	3	P2(12)
	C4	4	C3(12)
	C4	5	C3(8)
	C4	7	0V
	C4	8	B1(10)
	C4	10	C4(11)
	C4	12	C4(13),B2(6)
	C4	14	5V
XXXXXXXXXX	XXXXXX	XXXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXX
	B3	6	B2(10)
	B3	7	0V
	B3	8	B2(5)
	B3	14	5V
XXXXXXXXXX	XXXXXX	XXXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXX

IC INTERCONNECTION CHART

SIGNAL NAME	IC NO.	PIN	CONNECTED TO
	C3	1	B1(8)
	C3	7	0V
	C3	14	5V
XXXXXXXXXXXXX	XXXXXXX	XXXXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
	B2	7	0V
	B2	8	C2(13),C2(12)
	B2	14	5V
XXXXXXXXXXXXX	XXXXXXX	XXXXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
	C2	7	0V
	C2	8	B1(13)
	C2	10	C2(11)
	C2	14	5V
XXXXXXXXXXXXX	XXXXXXX	XXXXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
	B1	7	0V
	B1	11	0V
	B1	12	5V
	B1	14	5V
XXXXXXXXXXXXX	XXXXXXX	XXXXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
	C1	7	0V
	C1	14	5V
XXXXXXXXXXXXX	XXXXXXX	XXXXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
	R1	1	P2(5),µ47(-),via 4k7 to pad,R1(3)
			and µ47(+)
	R1	2	P1(88)
XXXXXXXXXXXXX	XXXXXXX	XXXXXXX	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

IC INTERCONNECTION CHART

APPENDIX II: MBM INTERFACE SOFTWARE

APPENDIX II: MBM INTERFACE SOFTWARE

Two separate programs are presented here. The first is the standard MBM interface software consisting of three subroutines for initialise, read and write. Rather than access just 1 page (18 bytes) at a time advantage was taken of the improved average access time if consecutive pages are accessed, by reading or writing 15 pages (270 bytes) at a time. The second is the MBM test program, which takes this idea still further by writing to the entire MBM in one go and reading the pages back sequentially for comparison. It was felt that this was necessary in order to reduce the duration of the test runs.

Quite simply the MBM interface software does three things:

i) Write command bytes
ii) Read status bytes
iii) Read and write data to and from the FIFO buffer in the controller

The details pertaining to these actions are contained in the data sheets produced by Texas Instruments for the TMS 5502. The two pages giving the broad details of the address functions, control and status bits are reproduced here.

TABLE 1. ADDRESS FUNCTIONS

ADDRESSES (A0–A3)	DATA COMMANDS	OPERATION
0	Load Page Select Reg 1	Write LSB
1	Load Page Select Reg 2	Write MSB
2	Load Control Command Register	Write (see Table 2)
3	Read Byte	Read FIFO
4	Write Byte	Write FIFO
5	Read Status	Read
6	Page Counter 1	Read/Write LSB
7	Page Counter 2	Read/Write MSB
8	Load Minor Loop Size Reg 1	Write LSB
9	Load Minor Loop Size Reg 2	Write MSB
A	Page Position Counter 1	Read/Write LSB
B	Page Position Counter 2	Read/Write MSB
C	Load Page Size Reg	Write
D	Read ROM Byte	Read & Incr. Pointer
E	Read/Write State Address	Read ROM ADDR Reg

NOTE: Address decode F will cause no action by the controller.

TABLE 2. CONTROL COMMAND BIT DEFINITION

BIT NO.	CONTROL COMMAND	RESET FUNCTION	FUNCTION
0	Initialize	System* + CRRST**	"Cold Start" initialization
1	Read Page	System + CRRST	Read page from bubble memory into FIFO page buffer
2	Write Page	System + CRRST	Write page from FIFO page buffer into bubble memory
3	Single Page Mode	Bit 4 = "1"	Sets up controller for single page transfers
4	Multi-Page Mode	System + Bit 3 = "1"	Sets up controller for multi-page transfers
5	Test Mode	System + Bit 5 = "0"	Enables control ROM verification
6	Reset		Software System Reset (Pulse)
7	Interrupt Mask	System + Bit 5 = "0"	Resets Interrupt Level (Pulse)

*System = Software Reset + Hardware Reset

**CRRST is a discrete control bit in the Control ROM.

2.2 STATUS REGISTER

Controller status is maintained in this register and can be read by the microprocessor. Table 3 lists these status conditions.

The following gives a detailed description of each bit:

Controller Busy – A logic "1" indicates that the controller is in the process of reading or writing data to the bubble module. A logic "0" indicates that the controller is in its idle state (State 0).

Load/Incr – A logic "1" indicates that the controller reached its present state by a branch condition. A logic "0" indicates the present state was reached sequentially from the previous state. This bit is updated on each state transition. (Used for testing.)

Inhibit – Indicates that the controller is in a state where transitions out are inhibited until the major loop shift counter goes to zero. (Used for testing).

Byte Read/Written – In the multi-page read mode, this bit indicates the FIFO buffer has been loaded with a byte of data and is ready for reading by the microprocessor. In the multi-page write mode, this bit indicates that the FIFO buffer has been transferred to the bubble data register and is ready to accept new data from the microprocessor. Byte RD/WR will be set every 160 μs (minimum). This status bit is reset when state is ready by the microprocessor.

Page Read/Written – This bit is utilized to indicate that a page of a data has been read into or written out of the FIFO buffer. This status bit is reset when status is read by the microprocessor. This bit is used for testing.

Page Count Zero – Set when the page counter counts to zero. Reset when the Page Count register is re-loaded. This bit indicates that the desired number of pages has been transferred by the controller.

Addressed Page Available – This bit indicates that the page select register is equal to the page position register. Status bit is reset when the status register is read by the microprocessor. This bit is used for testing.

NOTE

Since many of the status bits are reset on reading of the status register, there will be no conflict in setting and resetting the bits (i.e., the bit is set on a different clock phase than it is reset).

TABLE 3. STATUS CONDITIONS

BIT	CONDITION	SET FUNCTION	RESET FUNCTION
0	Page Read/Written	Page Size Counter	Status Read SRST
1	Load/INCR	Updated on State Transition	
2	Inhibit	Updated on State Transition	
3	MLSC Zero	Major Loop Shift Counter = 0	Status Read SRST
4	Addressed Page Available	Comparator	Status Read SRST
5	Controller Busy	Leaving State 0	"1" to "0" transition of BSS
6	Page Count Zero	Page Count Register	Page Count Register Loaded
7	Byte Read/Written	Byte Counter	Status Read SRST

```
#LOWER
       PAGE,CPAGE,CBYTE
INIT-      LDA ,*176       (LOAD MINOR LOOP SIZE TO 641)
           STO ,8200
           LDA ,*375
           STO ,8201
           LDA ,*355       (LOAD PAGE SIZE TO 18 BYTES)
           STO ,8204
           LDA ,*277
           STO ,8194
           LDA ,*376       (INITIALISE)
           STO ,8194
BY1-       LDA ,8197       (WAIT ON CONTROLLER BUSY)
           JBS 5 BY1
NBY1-      LDA ,8197
           JBC 5 NBY1
           RTC


WBLK-      LDA ,-15        (INITIALISE PAGE COUNT)
           STO CPAGE
WPAG-      LDA ,0          (INITIALISE PAGE NUMBER)
           STO PAGE
           LDA /1          (FETCH BLOCK NUMBER)
           ADS PAGE        (FIND FIRST PAGE BY MULTIPLYING
           SLA 1           (BLOCK NUMBER BY 15, STORE THE
           ADS PAGE        (RESULT IN LOCATION PAGE.)
           SLA 1
           ADS PAGE
           SLA 1
           ADS PAGE
           LDA /1+         (FETCH DATA AREA START ADDRESS)
           STO 190
PGWT-      LDA ,-18        (INITIALISE BYTE COUNT)
           STO CBYTE
           LDA PAGE        (FETCH PAGE NUMBER)
           NEQ ,*177777    (INVERT BITS)
           STO ,8192       (LOAD LS PAGE SELECT)
           SRA 8
           STO ,8193       (LOAD MS PAGE SELECT)
           LDA ,1          (INCREMENT PAGE NUMBER)
           ADS PAGE
           LDA ,*367       (SINGLE PAGE COMMAND)
           STO ,8194
FBYT-      LDA /190+       (FETCH DATA)
           STO ,8196       (WRITE TO FIFO)
           ICZ CBYTE FBYT  (18 BYTES ?)
           LDA ,*373       (WRITE PAGE COMMAND)
           STO ,8194
BY2-       LDA ,8197       (WAIT ON CONTROLLER BUSY)
           JBS 5 BY2
NBY2-      LDA ,8197
           JBC 5 NBY2
           ICZ CPAGE PGWT  (ALL PAGES WRITTEN ?)
           RTC
```

```
RBLK-   LDA ,-15          (INITIALISE PAGE COUNT)
        STO CPAGE
RPAG-   LDA ,0            (INITIALISE PAGE NUMBER)
        STO PAGE
        LDA /1            (FETCH BLOCK NUMBER)
        ADS PAGE          (FIND NUMBER OF FIRST PAGE BY MULTIPLING
        SLA 1             (BLOCK NUMBER BY 15, STORE IN LOCATION
        ADS PAGE          (PAGE)
        SLA 1
        ADS PAGE
        SLA 1
        ADS PAGE
        LDA /1+           (FETCH DATA START ADDRESS)
        STO 190
RDPG-   LDA ,-18          (INITIALISE BYTE COUNT)
        STO CBYTE
        LDA PAGE          (FETCH PAGE NUMBER)
        NEQ ,*177777      (INVERT BITS)
        STO ,8192         (LOAD LS PAGE SELECT)
        SRA 8
        STO ,8193         (LOAD MS PAGE SELECT)
        LDA ,1            (INCREMENT PAGE NUMBER)
        ADS PAGE
        LDA ,*357         (SINGLE PAGE COMMAND)
        STO ,8194
        LDA ,*375         (READ PAGE COMMAND)
        STO ,8194
BY3-    LDA ,8197         (WAIT ON CONTROLLER BUSY)
        JBS 5 BY3
NBY3-   LDA ,8197
        JBC 5 NBY3
RBYT-   LDA ,8195         (READ FROM FIFO)
        STO /190+         (STORE IN DATA AREA)
        ICZ CBYTE RBYT    (18 BYTES ?)
        ICZ CPAGE RDPG    (15 PAGES ?)
        RTC
```

```
$MASTER ERRORTEST

$MACRO MEMADDR
32512
$ENDMAC

$MACRO OUTPUT
        LDA ,DAREA-1
        STO 1
        CAL ,ZOUTCHAR
$ENDMAC

$COMMENT

*************************************************************************
*                                                                       *
*     PROGRAM TO TEST THE FREQUENCY AND TYPES OF ERRORS THAT            *
*     OCCUR WHEN USING MAGNETIC BUBBLE MEMORY. THE PROGRAM IS           *
*     WRITTEN IN SUCH A WAY THAT BY SIMPLY CHANGING THE VALUES          *
*     IN LOCATIONS 200 AND 201 THE STANDARD TEST PATTERNS OF            *
*     ALL ONE'S, ALL ZERO'S, AND CHECKERBOARD CAN BE USED.              *
*                                                                       *
*************************************************************************
$

$LOWER/200
        TPAT1,TPAT2,PT1/18,PT2/18,PAGE,CPAGE,CBYTE,PASS,NERROR
        FLAG,PATTER,MSCT,BITCT,DIGIT,HEXN,DAREA,WS/20
$LOWER
PR1-    33,=BYTE PATTERN FOR FIRST PAGE ?  - =
PR2-    33,=BYTE PATTERN FOR SECOND PAGE ? - =
PR3-    33,=INITIAL PASS NUMBER ?          - =
PR4-    29,=      PAGE    PASS    PATTERN=
PR5-     6,=TESTED=
PROMPT1- PR1
PROMPT2- PR2
PROMPT3- PR3
MESS1-   PR4
EMESS-   PR5
HLD24-    0,24
HLD30-    0,*2007
SUSP-  *300,FLAG-1
TTYOUT-   0
TTYIN-    1
DISKOUT-  2
TTOP-     0,0,TTYOUT
TTIP-     0,0,TTYIN
DKOP-     0,0,DISKOUT
$LOWER COMMON/CP
CP-     0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,31,63,127,255
        511,1023,2047,4095,8191,16383,32767,-1,CW-1,CW
CW-     /*11

$STACK/1535
$PROGRAM/2080
$START/2048
```

```
$COMMENT

      OPEN STREAMS 0,1 AND 2. THESE STREAMS ARE USED FOR :-
         0  -  OUTPUT TO TELYTYPE
         1  -  INPUT FROM TELETYPE
         2  -  OUTPUT TO DISK.
      THE BYTE PATTERNS TO BE LOADED INTO CONSECUTIVE PAGES IN
      THE MAGNETIC BUBBLE MEMORY ARE READ IN FROM THE TELETYPE
      ALONG WITH THE INITIAL VALUE TO BE LOADED INTO THE PASS
      COUNTER. THE RESULTS ARE OUTPUT TO DISK.
$
          LDA ,WS          (SET UP WORKSPACE STACK FOR USE BY F100
          STO 3            (LIBRARY ROUTINES.

          CAL ,ZINITIO     (INITIALISE I/O SOFTWARE PACKAGE.

          LDA ,TTOP-1      (OPEN STREAM 0 FOR TERMINAL OUTPUT.
          STO 1
          CAL ,ZOPENOUT
          LDA ,TTIP-1      (OPEN STREAM 1 FOR TERMINAL INPUT.
          STO 1
          CAL ,ZOPENINS
          LDA ,DKOP-1      (OPEN STREAM 2 FOR DISK FILE OUTPUT.
          STO 1
          CAL ,ZOPENOUT

IN1-      LDA ,TTYOUT-1    (SELECT STREAM 0 FOR OUTPUT TO TERMINAL.
          STO 1
          CAL ,ZSETOUTS

          LDA ,PROMPT1-1   (OUTPUT FIRST PROMPT.
          STO 1
          CAL ,ZOUTSTRI
          LDA ,TTYIN-1     (SELECT STREAM 1 FOR INPUT FROM TERMINAL.
          STO 1
          CAL ,ZSETINST
          CAL ,ZININT      (INPUT INTEGER TO ACCUMULATOR.
          STO TPAT1        (STORE IN TEST PATTERN LOCATION.
          CAL ,ZTESTSTA    (CHECK IF VALID INTEGER.
          JBS 15 IN1       (IF NON-NEGATIVE, OK.
          LDA ,*1000       (TEST IF INPUT INTEGER > *377.
          CMP TPAT1
          JBC S IN1


IN2-      LDA ,PROMPT2-1   (OUTPUT SECOND PROMPT.
          STO 1
          CAL ,ZOUTSTRI
          CAL ,ZININT      (INPUT INTEGER TO ACCUMULATOR.
          STO TPAT2        (AND STORE IN TEST PATTERN LOCATION.
          CAL ,ZTESTSTA    (CHECK IF VALID INTEGER.
          JBS 15 IN2       (IF NON-NEGATIVE, OK.
          LDA ,*1000       (TEST IF INPUT INTEGER > *377.
          CMP TPAT2
          JBC S IN2

IN3-      LDA ,PROMPT3-1   (OUTPUT THIRD PROMPT.
          STO 1
          CAL ,ZOUTSTRI
          CAL ,ZININT      (INPUT PASS NUMBER TO ACCUMULATOR
          STO PASS         (AND STORE IN PASS LOCATION.
          CAL ,ZTESTSTA    (CHECK IF VALID INTEGER.
          JBS 15 IN3
```

```
$COMMENT

      THIS CODE SETS THE 18 BYTES IN DATA AREA PT1 TO THE VALUE
      CONTAINED IN TPAT1, AND THE 18 BYTES IN DATA AREA PT2 TO
      THE VALUE CONTAINED IN TPAT2. PT1 AND PT2 ARE USED ON
      ALTERNATE PAGES IN THE BUBBLE MEMORY, FACILITATING A RANGE
      OF TEST PATTERNS.
$

          LDA ,-18          (LOAD BYTE COUNT INTO 191.)
          STO 191
          LDA ,PT1-1        (LOAD ADDRESS OF FIRST DATA AREA INTO 192.)
          STO 192
          LDA TPAT1         (LOAD FIRST TEST PATTERN INTO ACC.)
FILL-     STO /192+         (STORE TEST PATTERN IN DATA AREA.)
          ICZ 191 FILL      (PAGE COMPLETED?
          LDA ,-18          (IF SO, REPEAT PROCEDURE WITH TEST PATTERN
          STO 191           (IN TPAT2 AND DATA AREA PT2.)
          LDA ,PT2-1
          STO 192
          LDA TPAT2
PHIL-     STO /192+
          ICZ 191 PHIL
$COMMENT

      HAVING SET UP THE TEST PATTERN THE BUBBLE MEMORY UNIT MAY
      NOW BE TESTED.
$

          CAL ,INIT         (INITIALISE BUBBLE MEMORY UNIT.)

$COMMENT

      OUTPUT THE TABLE HEADINGS.
$
          LDA ,DISKOUT-1    (SELECT STREAM 2 FOR OUTPUT OF RESULTS
          STO 1             (TO A DISK FILE.
          CAL ,ZSETOUTS
          LDA ,MESS1-1      (SET UP PARAMETERS FOR LINE OUTPUT ROUTINE.
          STO 1
          CAL ,ZOUTSTRI     (CONTAINS 30 CHARACTERS.)
          LDA ,10           (OUTPUT NEWLINE.
          STO DAREA
          OUTPUT

          LDA ,-300         (SET MAXIMUM NUMBER OF ERRORS TO 300.)
          STO NERROR
$COMMENT

      TESTING BEGINS. THE TECHNIQUE USED IS TO WRITE THE TEST PATTERN
      TO THE ENTIRE BUBBLE MEMORY IN ONE GO, THEN READ BACK AND
      COMPARE PAGE BY PAGE.
$
STRT-     CAL ,WBLK         (WRITE PATTERN TO BUBBLE MEMORY.)
          LDA ,-641         (SET UP PAGE COUNT IN 191.)
          STO 191
          LDA ,0            (BEGIN AT PAGE 0.)
          STO PAGE
```

```
TEST-     JBS 0 PAGE PATT2(IF PAGE NUMBER IS EVEN THEN
          LDA ,PT1-1      (TEST PATTERN FOR PAGE IS CONTAINED
          JMP ,TSTPAT     (IN PT1, OTHERWISE
PATT2-    LDA ,PT2-1      (TEST PATTERN IS IN PT2, START
TSTPAT-   STO 190         (ADDRESS OF AREA IS IN 190.)
          CAL ,RBLK       (READ PAGE BACK FROM BUBBLE MEMORY.)
          LDA ,-18        (SET UP BYTE COUNT IN 193.)
          STO 193

COMP-     LDA ,3+MBMADDR  (FETCH BYTE FROM FIFO AND
          AND ,*377       (MASK OFF TOP BITS.)
          NEQ /190+       (COMPARE AGAINST TEST PATTERN.)

          JBS Z OK        (IF NO ERROR JUMP, ELSE
          STO PATTER      (SAVE ERROR PATTERN

          LDA ,= =        (OUTPUT SIX LEADING SPACES.
          STO DAREA
          LDA ,-6
          STO *100
SPACE-    OUTPUT
          ICZ *100 SPACE

          LDA PASS        (OUTPUT THE PASS NUMBER AS FOUR
          STO HEXN        (HEXADECIMAL DIGITS.
          CAL ,OHD

          LDA ,= =        (OUTPUT FOUR FORMATTING SPACES.
          STO DAREA
          LDA ,-4
          STO *100
BLKS-     OUTPUT
          ICZ *100 BLKS

          LDA PAGE        (THE PAGE NUMBER HAS BEEN INCREMENTED
          ADD ,-1         (AND IS THEREFORE DECREMENTED BEFORE
          STO HEXN        (BEING OUTPUT AS FOUR HEXADECIMAL
          CAL ,OHD        (DIGITS.

          LDA ,= =        (OUTPUT FOUR FORMATTING SPACES.
          STO DAREA
          LDA ,-4
          STO *100
SPS-      OUTPUT
          ICZ *100 SPS

          CAL ,ONEZER     (OUTPUT ERROR PATTERN AS BINARY.

          LDA ,10         (TERMINATE OUTPUT LINE WITH A NEWLINE.
          STO DAREA
          OUTPUT

          ICZ NERROR OK   (IF NUMBER OF ERRORS EXCEEDS 300 THEN
          CAL ,ZTERMIO
          LDA ,HLD24-1    (COME TO HOLD 24.
          STO 1
          CAL /23

OK-       ICZ 193 COMP    (END OF PAGE? IF SO IS THE
          LDA ,SUSP-1     (SUSPEND FLAG SET ?
```

```
            STO 1
            CAL /23
            JBC 0 FLAG CNTN
            CAL ,ZTERMIO
            LDA ,HLD30-1
            STO 1
            CAL /23
CNTN-       ICZ 191 TEST      (END OF PASS? IF SO
            LDA ,1            (ADD 1 TO PASS NUMBER.)
            ADS PASS
            JBC 5 STRT        (IF PASS NUMBER > 32K
            LDA ,EMESS-1
            STO 1
            CAL ,ZOUTSTRI
            LDA ,10
            STO DAREA
            OUTPUT
            CAL ,ZTERMIO
            LDA ,HLD24-1
            STO 1
            CAL /23

$COMMENT

      INITIALISE MAGNETIC BUBBLE MEMORY UNIT.
$
INIT-       LDA ,*176         (LOAD MINOR LOOP SIZE TO 641.)
            STO ,8+MBMADDR
            LDA ,*375
            STO ,9+MBMADDR
            LDA ,*355         (LOAD PAGE SIZE TO 18 BYTES.)
            STO ,12+MBMADDR
            LDA ,*277
            STO ,2+MBMADDR
            LDA ,*376         (INITIALISE.)
            STO ,2+MBMADDR
BY1-        LDA ,5+MBMADDR    (WAIT ON CONTROLLER BUSY.)
            JBS 5 BY1
NBY1-       LDA ,5+MBMADDR
            JBC 5 NBY1
            RTC

$COMMENT

      WRITE TEST PATTERN TO THE BUBBLE MEMORY.
$
WBLK-       LDA ,-641         (INITIALISE PAGE COUNT.)
            STO CPAGE
WPAG-       LDA ,0            (INITIALISE PAGE NUMBER.)
            STO PAGE
PGWT-       LDA ,-18          (INITIALISE BYTE COUNT.)
            STO CBYTE
            JBS 0 PAGE SECOND(IF PAGE NUMBER IS EVEN THEN
            LDA ,PT1-1        (WRITE THE PATTERN CONTAINED IN
            JMP ,SETPAT       (DATA AREA PT1 ELSE WRITE THE DATA
SECOND-     LDA ,PT2-1        (CONTAINED IN AREA PT2.)
SETPAT-     STO 190

            LDA PAGE          (FETCH PAGE NUMBER.)
            NEQ ,*177777      (INVERT BITS.)
```

```
            STO ,0+MBMADDR  (LOAD LS PAGE SELECT.)
            SRA 8
            STO ,1+MBMADDR  (LOAD MS PAGE SELECT.)
            LDA ,1          (INCREMENT PAGE NUMBER.)
            ADS PAGE
            LDA ,*367       (SINGLE PAGE COMMAND.)
            STO ,2+MBMADDR

FBYT-       LDA /190+       (FETCH DATA AND WRITE IT TO THE FIFO.)
            STO ,4+MBMADDR
            ICZ CBYTE FBYT  (16 BYTES?)

            LDA ,*379       (WRITE PAGE COMMAND.)
            STO ,2+MBMADDR
BY2-        LDA ,5+MBMADDR  (WAIT ON CONTROLLER BUSY.)
            JBS 5 BY2
NBY2-       LDA ,5+MBMADDR
            JBC 5 NBY2
            ICZ CPAGE PGWT  (ALL PAGES WRITTEN?)
            RTC


$COMMENT
      READ BACK DATA FROM BUBBLE MEMORY.
$
RBLK-       LDA PAGE        (FETCH PAGE NUMBER.)
            NEQ ,*177777    (INVERT BITS.)
            STO ,0+MBMADDR  (LOAD LS PAGE SELECT.)
            SRA 8
            STO ,1+MBMADDR  (LOAD MS PAGE SELECT.)
            LDA ,1          (INCREMENT PAGE NUMBER.)
            ADS PAGE
            LDA ,*365       (SINGLE PAGE READ COMMAND.)
            STO ,2+MBMADDR
BY3-        LDA ,5+MBMADDR  (WAIT ON CONTROLLER BUSY.)
            JBS 5 BY3
NBY3-       LDA ,5+MBMADDR
            JBC 5 NBY3
            RTC

$COMMENT
      OUTPUT ROUTINES ONEZER AND OHD SET UP DATA AREAS WHICH ARE
      SUBSEQUENTLY OUTPUT BY LINEOP. LINEOP OUTPUTS A LINE OF TEXT
      USING F100 I/O LIBRARY ROUTINES.
$

$COMMENT
      OUTPUT 1'S AND 0'S PATTERN.
$
ONEZER-     LDA ,-8         (SET UP BIT COUNT.)
            STO BITCT
TBIT-       JBS 7 PATTER ONE(IF MS BIT IS ZERO
            LDA ,=0=        (OUTPUT THE CHARACTER "0"
            JMP ,SHFT
ONE-        LDA ,=1=        (ELSE OUTPUT CHAPACTER "1".
SHFT-       STO DAREA
            OUTPUT
            SLL 1 PATTER
            ICZ BITCT TBIT  (8 BITS COMPLETED?)
            RTC
```

```
$COMMENT
      OUTPUT HEXADECIMAL NUMBER.
$
OHD-        LDA ,-4             (STORE DIGIT COUNT.
            STO DIGIT
NDIG-       LDA HEXN            (FETCH NUMBER TO BE OUTPUT.
            AND ,*170000        (COMPARE THE MS 4 BITS
            SRL 12              (WHEN MOVED TO LS 4 BITS
            CMP ,9              (WITH 9.
            JBC S NUMER         (IF GREATER THAN  9 ADD
            ADD ,7              (DECIMAL 55 TO MAKE UP ALPHABETIC
NUMER-      ADD ,48             (ELSE ADD 48 TO MAKE NUMERIC.
            STO DAREA
            OUTPUT
            SLL 4 HEXN          (LOOK AT NEXT 4 BITS.
            ICZ DIGIT NDIG      (4 DIGITS?)
            RTC

$END
$FINISH
```

APPENDIX III: CUMULATIVE AVERAGING PROGRAM

APPENDIX III: CUMULATIVE AVERAGING PROGRAM

The cumulative averaging program was written to extract the shape of a spike and wave complex from an EEG record. It relies on the spike and wave complexes having been already identified by the spike detector program developed by Dr. Richard Comley. False alarms were "de-marked" before the cumulative averaging program was run. As Dr. Comley's program sets bit 9 in the sample immediately following a spike each complex is not only identified but also aligned. It is, of course, necessary to store some samples prior to the sample with bit 9 set. Thus data is read in to a cyclic buffer of 16 words where the newest sample overwrites the oldest sample. When a sample with bit 9 set is encountered the cyclic buffer is stored with the ensuing 180 words to give a 196 sample EEG epoch containing a spike and wave complex. This epoch is then added to any previous epochs in a cumulative average build up area, and a count incremented. When enough epochs have been found to give a good average (up to 128), or the end of the EEG record is encountered, each location in the build up area is divided by the number of epochs found to give the cumulative average. The results are output graphically by using one of Dr. Comley's standard programs which outputs samples to a DAC for input to a y-t plotter.

```
 REL $CUMAV
SIZE    EQU  37          ;SIZE=CYC BUF LENGTH-1(MAX 77)
;;;;;;
;
;  PROGRAM TO EXTRACT SPIKE AND WAVE EPILEPSY
;  PRECURSOR FROM PREVIOUSLY PROCESSED EEG
;  RECORD. THE TECHNIQUE USED IS AS FOLLOWS:-
;   1/ A RECORD THAT HAS BEEN RUN THROUGH THE
;      SPIKE AND WAVE DETECTOR PROGRAM WRITTEN
;      BY R. COMLEY IS ASSUMED TO BE STORED ON
;      A STANDARD MAGNETIC DATA CASSETTE. THE
;      DATA IS NORMALLY 8 BITS WIDE, BUT THE
;      9TH BIT IS SET WHEN A SPIKE IS DETECTED.
;   2/ THE PROGRAM THEN RUNS A SMALL TRANSIENT
;      RECORDER ROUTINE TRIGGERED BY THE 9TH
;      BIT OF THE DATA BEING SET, I.E. A SPIKE
;      IS PRESENT.
;   3/ THE CAPTURED DATA  (WHICH CONSISTS OF 16
;      WORDS BEFORE AND 160 WORDS AFTER THE TRIGGER)
;      IS THEN SUMMED INTO A DATA AREA, AND A COUNT
;      OF THE NUMBER OF DETECTED SPIKES IS KEPT.
;      A MAXIMUM OF 127 SPIKES MAY BE USED WITHOUT
;      ANY DANGER OF ARITHMETIC OVERFLOW OCCURING.
;   4/ WHEN A SUFFICIENT NUMBER OF SPIKES HAVE
;      BEEN COLLECTED THEN THE TOTALS REACHED BY
;      SUMMING OF INDIVIDUAL RECORDS ARE DIVIDED
;      BY THE NUMBER OF SPIKES TO GIVE THE AVERAGE.
;
;;;;;;

R0      EQU  0
R1      EQU  1
R2      EQU  2
R3      EQU  3
R4      EQU  4
R5      EQU  5
SP      EQU  6
PC      EQU  7

 GLOB $CUMAV
$CUMAV CLRR R0
       MVII .256,R1   ;CLEAR CUMULATIVE AVERAGE
       MVII AVST,R5   ;BUILD UP AREA OF 200 WORDS.
CLEA   MVO@ R0,R5
       DECR R1
       BNZE CLEA
       MVII SPCT,R5   ;CLEAR THE FOLLOWING:
       MVO@ R0,R5     ;SPIKE COUNT REG (SPCT)
       MVO@ R0,R5     ;CYCLIC BUFFER PT (CBPT).
```

```
;;;;;;
;
;  INPUT DATA FROM CASSETTE AND STORE IN CYCLIC
;  BUFFER UNTIL TRIGGER OCCURS ON 9TH BIT.
;
;;;;;;
CASS    MVII 13,R0      ;SET CASSETTE CONTROL WORD
        MVO  R0,167742  ;FOR CONTINUOUS READ @14IPS.
WAIT    BEXT INPU,1     ;WAIT FOR IRDY.
        B    WAIT

;  INPUT DATA FROM CASSETTE.
INPU    MVO  R0,167744  ;ISSUE READ REQUEST.
HOLD    BEXT RDY,2      ;WAIT UNTIL DATA READY.
        B    HOLD
RDY     MVI  167745,R0  ;INPUT DATA.
        MVII CBST,R1    ;FETCH CYC BUF BASE ADDR.
        MVI  CBPT,R2    ;FETCH CYC BUF POINTER.
        ADDR R2,R1      ;ADD POINTER TO BASE.
        MVO@ R0,R1      ;STORE SAMPLE IN CYC BUF.

;  UPDATE CYCLIC BUFFER POINTER AND RECORD START
;  POINTER, WHICH IS 1 STEP AHEAD OF THE CBPT IN
;  THE CYCLIC BUFFER (I.E. THE FIRST WORD OF THE
;  LATEST INPUT DATA).
        INCR R2         ;POINTER=(POINTER+1)MOD16.
        ANDI SIZE,R2
        MVO  R2,CBPT    ;SAVE CYC BUF POINTER.

        ANDI 400,R0     ;TEST 9TH BIT OF DATA.
        BZE  INPU       ;IF NO TRIGGER, NEXT SAMPLE.

;;;;;;
;
;  AFTER A TRIGGER HAS BEEN DETECTED SAMPLES ARE
;  SUMMED INTO THE CUMULATIVE AVERAGE BUILD UP
;  AREA. THE START OF THE SPIKE AND WAVE IS IN
;  THE CYCLIC BUFFER BEGINNING AT THE POSITION
;  POINTED TO BY RCST. AFTER THE 16 WORDS OF THE
;  CYCLIC BUFFER ARE ADDED IN 160 SAMPLES (THOSE
;  FOLLOWING THE TRIGGER) ARE READ DIRECT FROM
;  THE CASSETTE AND SUMMED INTO THE DATA AREA.
;  FINALLY, A CHECK IS MADE TO ENSURE THAT THE
;  NUMBER OF SPIKES DOES NOT EXCEED THE MAXIMUM
;  OF 127.
;
;;;;;;
        MVII AVST,R5    ;SET UP CUM AVE DATA AREA.
        MVI  CBPT,R2    ;SET RECORD START ADDR.
XFER    MVII CBST,R1    ;FETCH CYC BUF BASE ADDR.
        ADDR R2,R1      ;ADD BUF OFFSET.
        MVI@ R1,R0      ;FETCH DATA...
        ANDI 377,R0     ;MASK OUT TOP BITS...
        ADD@ R5,R0      ;ADD IN TOTAL FROM BUILD UP...
        DECR R5         ;RESET R5 TO RIGHT SAMPLE...
        MVO@ R0,R5      ;AND STORE IN CUM AVE AREA.
        INCR R2         ;OFFSET=(OFFSET+1)MOD16.
        ANDI SIZE,R2
        CMP  CBPT,R2    ;CYC BUFFER EMPTIED ?
        BNZE XFER
```

```
;  ADD IN THE SAMPLES FOLLWING THE TRIGGER DIRECT
;  FROM CASSETTE TO CUMULATIVE AVERAGE BUILD UP
;  AREA.
        MVII .100,R3   ;SET UP SAMPLE COUNT
DATA    MVO  R0,167744;ISSUE READ REQUEST.
WWAT    BEXT OK,2      ;WAIT UNTIL DATA READY.
        B    WWAT
OK      MVI  167745,R0;INPUT DATA...
        ANDI 377,R0    ;MASK OUT TOP BITS...
        ADD@ R5,R0     ;ADD IN TOTAL FROM BUILD UP...
        DECR R5        ;RESET R5 TO RIGHT SAMPLE...
        MVO@ R0,R5     ;AND STORE IN CUM AVE AREA.
        DECR R3        ;POST TRIGGER COMPLETE ?
        BNZE DATA

;  CHECK IF NUMBER OF SPIKES EXCEEDS 126.
        MVI  SPCT,R3   ;INCREMENT SPIKE COUNT AND
        INCR R3        ;SEE IF MAXIMUM NUMBER IS
        MVO  R3,SPCT   ;EXCEEDED. IF SO THEN STOP
        CMPI .126,R3   ;PROCESSING.
        BMI  INPU

;;;;;;
;
;  HAVING SUMMED TOGETHER THE DETECTED SPIKES THE
;  AVERAGE IS NOW FOUND BY DIVIDING THESE TOTALS BY
;  THE TOTAL NUMBER OF SPIKES DETECTED. THIS IS
;  ACHIEVED BY USING SYSTEM ROUTINE $DIVI, WHICH IS
;  AN INTEGER DIVIDE ROUTINE.
;
;;;;;;
        MVII .256,R3   ;SET UP SAMPLE COUNT.
        MVII AVST,R5   ;POINT TO CUM AV DATA AREA.
        MVO  R5,LOCN   ;SAVE PRESENT SAMPLE ADDR.
AVER    CLRR R1        ;CLEAR LS HALF OF DIVIDEND.
        MVI  LOCN,R5   ;FETCH ADDR OF MS HALF.
        MVI@ R5,R2     ;LOAD MS HALF OF DIVIDEND.
        MVI  SPCT,R0   ;SET UP DIVISOR.
        JSR  R5,$DIVI  ;CALL DIVISION ROUTINE.
        TSTR R2        ;IF R2 IS NOT ZERO THEN THERE
        BNZE ERR       ;IS AN ERROR.
        MVI  LOCN,R5   ;POINT R5 TO PRESENT LOCATION.
        MVO@ R0,R5     ;SAVE CALCULATED VALUE.
        MVO  R5,LOCN   ;SAVE ADDR OF NEXT VALUE.
        DECR R3        ;256 SAMPLES ?
        BNZE AVER

ERR     HLT            ;HALT. A SUCCESSFUL RUN HAS R0
        HLT            ;CLEAR. IF R0>0, POS OV IN
        NOP            ;DIVV000ON IF R0<0 NEG OV.
        NOP
```

```
;;;;;;
;
;  INTEGER DIVIDE
;  CALLING SEQUENCE:
;   R0 = 16 BIT SIGNED DIVISOR
;   R1,R2 = 32 BIT SIGNED DIVIDEND
;   JSR R5,£DIVI
;  NORMAL EXIT:
;   R0 = 16 BIT SIGNED QUOTENT
;   R1 = 16 BIT SIGNED REMAINDER
;   R2 = ZERO
;   R3,R4 UNCHANGED
;   R5 = ABS VALUE OFF DIVISOR
;  ERROR EXIT:

;   R0 = ABS VALUE OF DIVBISOR
;   R1 = ABS VALUE OF MS HALF DIVIDEND
;   R2 = NON ZERO
;    R2 > 0, POS OVERFLOW
;    R2 < 0, NEG OVERFLOW
;   R3,R4 UNCHANGED
;   R5 = ABS VALUE OF DIVISOR
;
;;;;;;
£DIVI PSHR R5 ;SAVE RETURN
 PSHR R3 ;SAVE R3
 MVII .16,R3 ;LOOP COUNT
 TSTR R0 ;CHK DIVISOR SIGN
 BPL £DIVI1 ;POS
 NEGR R0 ;NEG, NEG DIVISOR
 COMR R5 ;COMPLEMENT SIGN INDICATOR
£DIVI1 PSHR R0 ;SAVE ABS DIVISOR
 TSTR R1 ;CHK DIVIDEND SIGN
 BPL £DIVI2 ;POS
 SLR R2,1 ;NEG, CLR DIVIDE3ND LSB
 SLL R2,1
 COMR R1 ;NEG DIVIDEND
 NEGR R2
 ADCR R1
 BOV £DIVI6 ;LARGEST NEG NO.
 COMR R5 ;COMPLEMENT SIGN INDICATOR
£DIVI2 CMPR R0,R1 ;CHK FOR ERROR
 BLGE £DIVI6 ;DIVIDEND .GE. DIVISOR
 ADDR R2,R2
£DIVI3 RLC R1,1 ;SHFT DIVIDEND
 CMPR R0,R1 ;SUBTRACTION POSSIBLE ?
 BLLT £DIVI4 ;NO
 SUBR R0,R1 ;YES, SUBTRACT OUT DIVISOR
£DIVI4 RLC R2,1
 DECR R3 ;LOOP FINI ?
 BNZE £DIVI3 ;NO
 MOVR R2,R0 ;YES, QUIOTENT TO R0
 MOVR R3,R2 ;ERR FLG= 0
```

```
 INCR PC ;SKP NXT INSTR
£DIVI6 MOVR R5,R2 ;ERROR, SIGN INDICATOR TO R2
 TSTR R5 ;CHK FOR NEG REQUIRED
 BPL £DIVI5 ;POS, OK
 NEGR R0 ;NEG QUOTENT
 NEGR R1 ;NEGATE REMAINDER
£DIVI5 PULR R5 ;ABS DIVISOR
 PULR R3 ;RESTORE R3
 PULR PC ;EXIT

SPCT    RES  .1
CBPT    RES .1
CBST    RES  .64
AVST    RES  .256
LOCN    RES  .1
 END
```

APPENDIX IV: M68000 FILTER PROCESS BENCHMARK

APPENDIX IV: M68000 FILTER PROCESS BENCHMARK

The program given here is purely for the purpose of estimating the improvement in performance of the generalised signal detector should the microprocessor used be changed from a F100-L to a M68000. As no development equipment for the M68000 was available the program was never assembled and run, and there is no guarantee that it would, in fact, run correctly. However, this is perhaps academic as the purpose of the exercise was to gain some feel of a "ball-park" figure for performance improvement.

```
START   MOVE    STAT,A2         ;WAIT UNTIL THE ADC STATUS
BACK    BTST    15,A2@          ;WORD INDICATES IT IS READY
        BZE     BACK            ;FOR INPUT AND OUTPUT.
        MOVE    R3,DATA         ;OUTPUT SAMPLE THEN
        MOVE    DATA,R0         ;INPUT SAMPLE BYTE INTO
        MOVE    R0,A0@(A1.L)    ;CYCLIC BUFFER.
        ADDQ    #2,A0           ;MAINTAIN CYCLIC BUFFER POINTER
        CMPI    #128,A0         ;FOR 128 BYTES.
        BPL     OVER
        CLR     A0
OVER    MOVE    #64,R2          ;SET COUNT FOR 64 COEFFICIENTS.
        MOVE    IR-1,A3         ;FETCH THE IMPULSE RESPONSE.
        CLR     R3
RET     ADDQ    #2,A0           ;POINT TO SAMPLE IN CYCLIC BUFFER
        CMPI    #128,A0         ;AND MAINTAIN CYCLIC BUFFER
        BPL     PASS            ;POINTER.
        CLR     A0
PASS    MOVE    A0@(A1.L),R0    ;PUT SAMPLE INTO R0
        MOVE    A3@+,R1         ;AND COEFFICIENT INTO R1
        MULU    R0,R1           ;THEN MULTIPLY THEM AND ADD
        ADD     R0,R3           ;TO THE OUTPUT REGISTER R3.
        DBNZ    R2   RET        ;64 SAMPLES/COEFFICIENTS?
        BR      START
```

APPENDIX V: GENERALISED SIGNAL DETECTOR PROTOTYPE SOFTWARE

APPENDIX V: GENERALISED SIGNAL DETECTOR PROTOTYPE SOFTWARE

The following pages contain the source code listing for the generalised signal detector prototype software. A detailed description of the program is given in Chapter 9.

```
$MASTER OPTIMUMFILTER
$COMMENT
*****************************************************************
*                                                               *
*   THIS PROGRAM PERFORMS THREE TASKS, DEPENDING ON THE CHARACTER *
*   INPUT TO IT. IF THE CHARACTER "A" IS TYPED, THEN THE PROGRAM  *
*   PERFORMS AN AUTOCORRELATION ON THE SIGNAL INPUT THROUGH THE   *
*   ANALOG-TO-DIGITAL CONVERTER. THE AUTOCORRELATION CONSISTS OF  *
*   64 POINTS GENERATED FROM 8192 INPUT SAMPLES.                  *
*   IF THE CHARACTER "O" IS TYPED THE PROGRAM USES THE AUTOCOR-   *
*   RELATION TO DERIVE THE IMPULSE RESPONSE OF THE FILTER WITH    *
*   THE OPTIMUM SIGNAL-TO-NOISE RATIO FOR THE GIVEN INPUT NOISE   *
*   TO DETECT A SIGNAL DEFINED BY AREA IR.                        *
*   IF THE CHARACTER "F" IS TYPED, THEN THE PROGRAM IMPLEMENTS    *
*   THE FILTER WHOSE PARAMETERS ARE CALCULATED BY THE OPTIMISATION*
*   ROUTINE.                                                      *
*                                                               *
*****************************************************************
$
$LOWER
        CALC/8,UTU/2,BUFF/64,WS/20,SCA,NOFP
$LOWER
ALPHA-   *020000
TTYI-     1
TTIP-     0,0,TTYI
HLD17-   0,17
COND-     64,IR-1
APAR-     0
OPSO-    *500
PPAO-    *000,*0
CPSO-    *300
APOI-    *103,*0
OASI-    *503
CASI-    *303
FPAO-    *004,*0
FPAI-    *105,*0
OPAO-    *504
OPAI-    *505
CPAO-    *304
CPAI-    *305
SPAR-     RXX-1,-64,9
$LOWER COMMON/PARAM
CBPA-     BUFF,64,0
CBPTW-    0
$LOWER COMMON/CP
CP-       0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,31,63,127,255,511
          1023,2047,4095,8191,16383,32767,-1,CW-1,CW
CW-       /*11
$UPPER
          RXX/128,SQTAB/256,PLOT/9000
$UPPER
SIG-      *31200,*31400,*31400,*32000,*32200,*32600,*33200,*33200
          *33200,*33400,*33600,*34000,*34000,*34200,*34600,*35200
          *35600,*36000,*36200,*36400,*37000,*37200,*37600,*37600
          *40200,*40400,*40600,*40600,*40600,*40400,*40400,*40400
          *40400,*40600,*40200,*37400,*37000,*36400,*36000,*34000
          *35000,*34200,*33000,*31600,*30200,*27000,*25200,*24200
          *23000,*21600,*21200,*20200,*20200,*20000,*22200,*27200
          *36200,*44600,*50200,*46000,*37600,*33200,*31600,*32200
IR-       *200/64
```

```
$PROGRAM
$START/2048
        LDA ,WS             (SET UP WORKSPACE STACK FOR LIBRARY
        STO 3               (ROUTINES.
        CAL ,ZINITIO
        LDA CP+0            (CREATE THE TABLE OF SQUARES BY USING
        STO ,SQTAB          (THE FORMULA [I+1]**2=[I]**2+2I+1.
        LDA CP+1            (THAT IS TO SAY GENERATE THE NEXT NUMBER
        STO *100            (IN THE SEQUENCE BY ADDING THE NEXT ODD
        LDA ,SQTAB          (NUMBER TO THE PREVIOUS SQUARE.
        STO *102
RPTS-   LDA /*102
        ADD *100
        JBS C BEG
        STO /*102+
        LDA CP+2
        ADS *100
        JMP ,RPTS
BEG-    LDA ,-63            (INITIALISE AUTOCORRELATION TO
        STO *100            (THE VALUES RXX[0]=255**2.
        LDA ,RXX-1          (ALL OTHERS=1024.
        STO *101
        LDA ,*77771
        STO /*101+
        LDA ,0
        STO /*101+
BRXX-   LDA ,1024
        STO /*101+
        LDA ,0
        STO /*101+
        ICZ *100 BRXX
        LDA ,TTIP-1         (OPEN INPUT STREAM 1.
        STO 1
        CAL ,ZOPENINS
        LDA ,TTYI-1         (SELECT STREAM 1 FOR INPUT.
        STO 1
        CAL ,ZSETINST
TASK-   CAL ,ZINCHAR        (INPUT A CHARACTER.
        CMP ,=E=            (IF AN "E" WAS TYPED THEN END ELSE
        JBS Z STOP
        CMP ,=A=            (IF AN "A" WAS TYPED THEN GOTO THE
        JBS Z AUTAR         (AUTOCORRELATION ROUTINE ELSE
        CMP ,=O=            (IF AN "O" WAS TYPED THEN GOTO THE
        JBS Z OPTI          (PARAMETER FINDING ROUTINE.
        CMP ,=F=            (IF AN "F" WAS TYPED THEN GOTO THE
        JBS Z IOFI          (FILTERING ROUTINE.
        CMP ,=P=            (IF A "P" WAS TYPED THEN LOAD
        JBS Z POR           (OUTPUT AREA TO STREAM 0.
        JMP ,TASK           (IF NONE OF THESE, INPUT ANOTHER.

$COMMENT
        PERFORM AUTOCORRELATION ON INPUT SIGNAL.
$
AUTAR-  LDA ,-128           (CLEAR THE DATA AREAS RXX.
        STO *100
        LDA ,RXX-1
        STO *101
        LDA ,0
CLRR-   STO /*101+
        ICZ *100 CLRR
```

```
        LDA ,OASI-1       (OPEN BINARY STREAM 3.
        STO 1
        CAL /21
        LDA ,-63          (INITIALISE BUFF BY LOADING IT UP
        STO *100          (WITH SAMPLES X[-63] TO X[-1].
LUBU-   LDA ,APOI-1       (INPUT A SAMPLE FROM STREAM 3.
        STO 1
        CAL /21
        LDA APOI+1
        AND ,*377         (MASK OFF AND INVERT THE LEAST
        NEQ ,*377         (SIGNIFICANT 8 BITS.
        STO CBPA+2        (PLACE SAMPLE IN CYCLIC BUFFER.
        CAL ,IPCB
        ICZ *100 LUBU
        LDA ,-8192        (CALCULATE THE AUTOCORRELATION OVER 8192
        STO *100          (SAMPLES.
IPSM-   LDA ,APOI-1       (INPUT DATA ON STREAM 3.
        STO 1
        CAL /21
        LDA APOI+1
        AND ,*377
        NEQ ,*377
        STO *101          (SAVE CURRENT SAMPLE IN *101 AND ALSO
        STO CBPA+2        (PLACE IT IN THE CYCLIC BUFFER.
        CAL ,IPCB
        LDA ,-64          (MULTIPLY THE LATEST SAMPLE BY ITSELF AND
        STO *102          (THE PREVIOUS 63 SAMPLES.
        LDA ,RXX-1        (THE 64 PRODUCTS ARE SUMMED IN ORDER
        STO *103          (INTO THE DATA AREA RXX.
CARY-   CAL ,OPCB         (FETCH APPROPRIATE SAMPLE INTO CBPA+2 AND
        CAL ,MULT         (MULTIPLY BY LATEST SAMPLE AND ADD.
        LDA /*103+        (MOVE POINTER ONTO LS HALF OF AUTOC.
        LDA /*103+        (FETCH LS HALF OF AUTOCORRELATION AND
        ADD *104          (ADD IN PRODUCT.
        SET M             (M STAT SET FOR DOUBLE LENGTH ADDITION.
        STO /*103-
        LDA ,0            (ADD ANY CARRY INTO MS HALF OF THE
        ADD /*103         (AUTOCORRELATION.
        CLR M
        STO /*103
        LDA /*103+        (MOVE POINTER ON TO NEXT POINT.
        ICZ *102 CARY     (64 POINTS COMPLETED ?
        ICZ *100 IPSM     (8192 SAMPLES COMPLETED ?
        LDA ,CASI-1       (CLOSE STREAM 3.
        STO 1
        CAL /21
        LDA ,-63
        STO *101
        LDA ,RXX+127
        STO *100
        LDA /*100-
        STO *103
        LDA /*100-
        STO *102
LL1-    LDA /*100-
        CMP *103
        SET M
        LDA /*100-
        CMP *102
        CLR M
```

```
        JBS S LL2
        LDA /*100+
        STO *102
        LDA /*100+
        STO *103
        LDA /*100-
        LDA /*100-
LL2-    ICZ *101 LL1
        LDA ,-64
        STO *101
        LDA ,RXX+127
        STO *100
SFRO-   CLR M
        LDA *103
        SBS /*100-
        SET M
        LDA *102
        SBS /*100-
        ICZ *101 SFRO
        LDA ,RXX-1          (NORMALISE R[0].
        STO 1
        CAL ,SFAC
        LDA ,RXX+1          (SHIFT ALL THE OTHER POINTS IN THE
        STO *125            (AUTOCORRELATION BY AN AMOUNT
        STO *130            (DETERMINED BY THE SCALE FACTOR
        LDA ,-63            (RETURNED FROM THE SCALING ROUTINE.
        STO *131
SACL-   LDA *123            (FETCH THE CALCULATED SCALE FACTOR AND
        CMP ,0              (CHECK IT IS NON-ZERO.
        JBS Z MORE          (IF IT IS NON-ZERO, THEN
        NEQ ,*177777        (NEGATE IT IN ORDER TO MAKE A COUNT.
        ADD ,1
        STO *124
        LDA /*125+          (FETCH THE MOST SIGNIFICANT HALF OF THE
        STO *126            (AUTOCORRELATION.
        LDA /*125+          (FETCH THE LEAST SIGNIFICANT HALF OF
        STO *127            (THE AUTOCORRELATION.
        LDA *126
        SET M               (DO A DOUBLE LENGTH LEFT SHIFT UNTIL
CONS-   SLL 1 *127          (THE COUNT IS ZERO.
        ICZ *124 CONS
        CLR M
        STO /*130+          (RESTORE THE SHIFTED AUTOCORRELATION.
        LDA *127
        STO /*130+
        ICZ *131 SACL
MORE-   LDA ,-64            (OUTPUT 64 POINTS TO PLOTTER AREA.
        STO *100
        STO NOFP
        LDA ,RXX-1
        STO *101
        LDA ,PLOT-1         (INITIALISE PLOTTER AREA POINTER.
        STO *131
HOLD-   LDA /*101+          (FETCH DOUBLE LENGTH AUTOCORRELATION.
        STO /*131+
        LDA /*101+          (MOVE ON POINTER.
        ICZ *100 HOLD
        JMP .TASK
MULT-   LDA *104            (CLEAR PRODUCT REC FOR 8X8 BIT
        SBS *104            (MULTIPLY.
```

```
        LDA ,-8             (INITIALISE COUNT.
        STO *105
        LDA *101            (FETCH LATEST SAMPLE.
LOOP-   JBC 0 CBPA+2 ZERO(LS BIT OF MPLIER SET ?
        ADS *104            (IF SO ADD IN MCAND ELSE
ZERO-   SLL 1               (SHIFT MCAND LEFT AND MPLIER
        SRA 1 CBPA+2        (RIGHT.
        ICZ *105 LOOP
        RTC
IOFI-   LDA ,0              (CLEAR THE DOUBLE LENGTH OUTPUT
        STO *111            (LOCATIONS.
        STO *112
        LDA ,IR-1
        STO *100
        LDA ,-63
        STO *101
        LDA /*100+
        STO *102
LOP1-   LDA /*100+
        CMP *102
        JBC S LOP2
        STO *102
LOP2-   ICZ *101 LOP1
        LDA ,0
        STO *100
        JBS 14 *102 LOP6
LOP3-   SLL 1 *102
        LDA ,-1
        ADS *100
        JBC 14 *102 LOP3
LOP4-   LDA ,-64
        STO *101
        LDA ,IR-1
        STO *103
LOP5-   LDA /*103+
        SLL 1
        STO /*103
        ICZ *101 LOP5
        ICZ *100 LOP4
LOP6-   LDA ,OPAO-1         (OPEN BINARY STREAM 4.
        STO 1
        CAL /21
        LDA ,OPAI-1          (OPEN BINARY STREAM 5.
        STO 1
        CAL /21
        LDA ,FPAI-1         (INPUT THE FIRST SAMPLE ON STREAM
        STO 1               (4 AND LOAD IT INTO EACH OF THE
        CAL /21             (64 LOCATIONS IN THE CYCLIC
        LDA ,-64            (BUFFER. I.E. SET UP THE VALUE
        STO *104            (OF THE FIRST SAMPLE AS BEING THE
        LDA ,BUFF-1         (ZERO OR STEADY STATE VALUE.
        STO *103
        LDA FPAI+1
ICB-    STO /*103+
        ICZ *104 ICB
NSAM-   LDA *111            (OUTPUT FILTERED SAMPLE.
        SET M
        SRA 5 *112
        CLR M
        NEQ ,*377
```

```
        STO FPAO+1
        LDA ,FPAO-1
        STO 1
        CAL /21
        LDA ,FPAI-1        (FETCH SAMPLE ON STREAM 5.
        STO 1
        CAL /21
        LDA FPAI+1
        JBS 15 DOBE        (EOF ?
        AND ,*377
        NEQ ,*377
        STO CBPA+2
        CAL ,IPCB
        LDA ,COND-1
        STO 1
        CAL ,FILT
        JMP ,NSAM
DOBE-   LDA ,CPAO-1        (CLOSE OUTPUT STREAM.
        STO 1
        CAL /21
        LDA ,CPAI-1         (CLOSE INPUT STREAM.
        STO 1
        CAL /21
        JMP ,TASK
STOP-   CAL ,ZTERMIO        (PREPARE FOR HOLD.
        LDA ,HLD17-1
        STO 1
        CAL /23

$COMMENT
        FIND THE FILTER PARAMETERS USING SEQUENTIAL LEARNING
        IDENTIFICATION.
$
OPTI-   LDA ,RXX+127            (FETCH ADDRESS OF RXX[63] AND
        STO *100                (STORE IN A POINTER.
        LDA ,-63                (SET UP COUNTER.
        STO *101
REV-    LDA /*100-              (FETCH AUTOCORRELATION IN
        LDA /*100-              (REVERSE ORDER INTO BUFFER.
        STO CBPA+2
        CAL ,IPCB
        ICZ *101 REV
        LDA ,-64                (SET UP COUNT FOR OUTER LOOP.
        STO *100
        LDA ,RXX-1              (SET UP AUTOCORRELATION
        STO *101                (POINTER.
        LDA ,SIG-1              (SET UP POINTER TO THE AREA
        STO *102                (CONTAINING SIGNAL.
OBEG-   LDA ,29                 (INITIALISE THE SCALE FACTOR.
        STO *130
        LDA ALPHA               (FETCH ERROR CORRECTING
        STO CALC+0              (COEFFICIENT.
        LDA /*101+              (FETCH NEXT POINT IN THE
        STO CBPA+2              (AUTOCORRELATION INTO THE
        CAL ,IPCB               (BUFFER.
        LDA /*101+              (MOVE ON POINTER.
        LDA ,COND-1             (PERFORM CONVOLUTION FUNCTION.
        STO 1
        CAL ,FILT
```

```
        LDA *111            (FETCH FILTER OUTPUT AND
        SET M               (MOVE IT TO THE TOP OF THE WORD.
        SLA 2 *112
        STO *111
        CLR M
        SUB /*102+          (SUBTRACT FROM SIGNAL.
        STO CALC+1          (MULTIPLY DIFFERENCE BY THE
        LDA ,CALC-1         (ERROR CORRECTING COEFFICIENT.
        STO 1
        CAL ,ZMULT
        LDA ,CALC+1         (SCALE THE DOUBLE LENGTH
        STO 1               (PRODUCT.
        CAL ,SFAC
        LDA *123            (UPDATE THE SCALE FACTOR.
        ADS *130
        LDA CALC+2          (NORMALISED VALUE IS SHIFTED
        SET M               (RIGHT TWO PLACES SO THAT IT
        SRA 2 CALC+3        (IS LESS THAN THE DIVISOR.
        CLR M               (ENSURING NO OVERFLOW IN THE
        STO CALC+2          (DIVISION OPERATION.
        LDA ,-64            (CALCULATE UTU BY THE USE OF
        STO *103            (LOOK-UP TABLE.
        LDA ,BUFF-1         (POINT TO BUFFER CONTAINING
        STO *104            (THE INPUT VECTOR.
        LDA ,0              (CLEAR UTU REGS.
        STO UTU
        STO UTU+1
SQ-     LDA /*104+          (FETCH INPUT, TAKE THE M.S.
        SRL 7               (8 BITS AND ADD TO BASE OF
        ADD ,SQTAB          (LOOK-UP TABLE. USE THIS AS
        STO *170            (THE ADDRESS OF THE VALUE OF
        LDA /*170           (SQUARE OF INPUT AND ADD IT
        ADD UTU+1           (DOUBLE LENGTH INTO THE
        SET M               (CURRENT TOTAL.
        STO UTU+1
        LDA ,0
        ADD UTU
        CLR M
        STO UTU
        ICZ *103 SQ         (REPEAT 64 TIMES.
        LDA ,UTU-1          (WHEN COMPLETE NORMALISE THE
        STO 1               (RESULT.
        CAL ,SFAC
        LDA *123            (UPDATE SCALE FACTOR.
        SBS *130
        LDA UTU             (FETCH NORMALISED DIVISOR.
        STO CALC+1
        LDA ,CALC+0         (PERFORM DIVISION.
        STO 1
        CAL ,ZDIV
        LDA ,-64            (SET UP COUNT FOR INNER LOOP.
        STO *104
        LDA ,IR-1           (SET UP IMPULSE RESPONSE
        STO *106            (POINTER.
REPEAT- CAL ,OPCB           (FETCH THE INPUT VECTOR.
        LDA CBPA+2          (LATEST SAMPLE FIRST.
        STO CALC+5
        LDA ,CALC+3         (MULTIPLY BY INTERMEDIATE
        STO 1               (RESULT.
        CAL ,ZMULT
```

```
        LDA *130                (PLACE SCALE FACTOR IN
        STO SCA                 (TEMPORARY LOCATION.
RFK-    LDA SCA                 (DENORMALISE RESULT. IF SCALE
        CMP ,0                  (FACTOR IS ZERO THEN COMPLETE
        JBC S OUT               (ELSE...
        LDA CALC+6              (SHIFT RESULT ONE PLACE RIGHT
        SET M                   (AND DECREMENT THE SCALE
        SRA 1 CALC+7            (FACTOR.
        CLR M
        STO CALC+6
        LDA ,-1
        ADS SCA
        JMP .RFK
OUT-    LDA CALC+7              (FETCH DENORMALISED RESULT AND
        ADS /*106+              (ADD TO IMPULSE RESPONSE.
        ICZ *104 REPEAT         (64 PARAMETERS ?
        ICZ *100 OBEG           (64 ITERATIONS ?
AGIN-   LDA SPAR+1
        STO *104
        STO NOFP
        LDA COND+1
        STO *106
        LDA ,PLOT-1
        STO *100
TIME-   LDA /*106+
        STO /*100+
        ICZ *104 TIME
        JMP .TASK
$COMMENT
        OUTPUT PLOTTER AREA TO STREAM 0 AFTER SUBTRACTING
        THE SMALLEST VALUE FROM EACH ORDINATE AND THEN
        NORMALISING.
$
POR-    LDA .PLOT-1             (SET UP PLOTTER AREA POINTER.
        STO *100
        LDA NOFP                (FETCH NUMBER OF POINTS TO BE
        ADD ,-1
        STO *101                (OUTPUT.
        LDA /*100+              (INITALISE "LARGEST SO FAR".
        STO *102
        STO *103
BORS-   LDA /*100+              (FETCH NEXT ORDINATE.
        CMP *102                (IF IT IS LARGER THAN THE
        JBC S STSF              ("LARGEST SO FAR" THEN UPDATE
        STO *102                (LOCATION *102.
STSF-   CMP *103                (IF IT IS SMALLER THAN THE
        JBS S COMP              ("SMALLEST SO FAR" THEN UPDATE
        STO *103                (LOCATION *103.
COMP-   ICZ *101 BORS
        LDA .PLOT-1             (SET UP POINTER TO START OF
        STO *100                (DATA AREA IN ORDER TO SUBTRACT
        LDA NOFP                (THE SMALLEST VALUE FOUND FROM
        STO *101                (EACH ORDINATE.
        LDA *103                (FETCH SMALLEST AND SUBTRACT
        SBS *102                (FROM THE LARCEST VALUE AS WELL
SSFO-   SBS /*100+              (FROM EVERY ORDINATE.
        ICZ *101 SSFO
        LDA *100                (CLEAR LOCATION *100 IN ORDER
        SBS *100                (TO BUILD UP NORMALISATION COUNT.
        LDA ,0
```

```
        CMP *102
        JBS Z BTOO
        LDA ,1
IINV-   SLL 1 *102          (NORMALISE LARGEST VALUE BY
        ADS *100            (SHIFTING UNTIL BIT 15 IS SET,
        JBC 15 *102 IINV    (INCREMENTING *100 EACH TIME.
        LDA *100            (FETCH THE NUMBER OF SHIFTS AND
        NEQ ,*177777        (TAKE THE TWO'S COMPLEMENT IN
        ADD ,1              (ORDER TO MAKE ACOUNT.
        STO *104
        LDA ,PLOT-1         (SHIFT EVERY ORDINATE BY THE
        STO *100            (AMOUNT HELD IN THE COUNT.
        STO *103
        LDA NOFP            (INITIALISE NUMBER OF POINTS
        STO *101            (COUNT.
SNO-    LDA *104            (FETCH SHIFT COUNT.
        STO *102
        LDA /*100+          (FETCH THE ORDINATE AND SHIFT
NTO-    SLL 1               (LEFT UNTIL THE COUNT IS ZERO.
        ICZ *102 NTO
        SRL 8               (WHEN FINISHED TAKE THE M.S.
        STO /*103+          (8 BITS AND RESTORE TO AREA.
        ICZ *101 SNO
BTOO-   LDA ,OPSO-1         (OPEN BINARY STREAM 0.
        STO 1
        CAL /21
        LDA ,PLOT-1         (OUTPUT THE ORDINATES TO STREAM.
        STO *100
        LDA NOFP
        STO *101
OOTS-   LDA /*100+          (FETCH THE ORDINATE AND OUTPUT
        NEQ ,*377           (THE INVERTED VALUE TO BINARY
        STO PPAO+1          (STREAM 0.
        LDA ,PPAO-1
        STO 1
        CAL /21
        ICZ *101 OOTS
        LDA ,CPSO-1         (CLOSE BINARY STREAM 0.
        STO 1
        CAL /21
        JMP ,TASK

$END

$SEGMENT INPUTCYCLICBUFFER
$COMMENT
       THE PARAMETERS FOR THIS ROUTINE ARE PASSED OVER IN A
       COMMON BLOCK LABELLED "PARAM". THE PARAMETERS ARE :-
        +0      BASE ADDRESS OF CYCLIC BUFFER.
        +1      LENGTH OF CYCLIC BUFFER.
        +2      SAMPLE TO BE WRITTEN.
        +3      CYCLIC BUFFER WRITE POINTER.
        +4      CYCLIC BUFFER READ POINTER.
$
$LOWER COMMON/PARAM
       CBPA/3,CBPTW
$PROGRAM
$ENTRY
IPCB-  LDA CBPA+0
       ADD CBPTW
```

```
        STO *105
        LDA CBPA+2
        STO /*105
        LDA CBPTW
        ADD ,1
        CMP CBPA+1
        JBC Z NOV
        LDA ,0
NOV-    STO CBPTW
        RTC
$END


$SEGMENT OUTPUTCYCLICBUFFER
$COMMENT
        THE PARAMETERS ARE TRANSFERRED IN A COMMON BLOCK
        CALLED "PARAM". THE PARAMETERS ARE :-
         +0     CYCLIC BUFFER BASE ADDRESS.
         +1     LENGTH OF CYCLIC BUFFER.
         +2     SAMPLE READ BY ROUTINE.
         +3     CYCLIC BUFFER WRITE POINTER.
         +4     CYCLIC BUFFER READ POINTER.
$
$LOWER COMMON/PARAM
       CBPA/3,CBPTW
$PROGRAM
$ENTRY
OPCB-   LDA CBPTW
        ADD ,-1
        JBC S NOV
        LDA CBPA+1
        ADD ,-1
NOV-    STO CBPTW
        ADD CBPA+0
        STO *105
        LDA /*105
        STO CBPA+2
        RTC
$END


$SEGMENT CONVOLUTION
$COMMENT
        THIS SUBROUTINE PERFORMS A DIGITAL FILTER FUNCTION
        BY CONVOLVING THE INPUT SAMPLES WITH THE IMPULSE
        RESPONSE OF THE DESIRED FILTER. THE SAMPLES ARE
        ASSUMED TO BE IN A CYCLIC BUFFER. THE PARAMETERS
        FOR THIS ROUTINE ARE :-
         +0     THE NUMBER OF COEFFICIENTS IN THE IMPULSE
                RESPONSE.
         +1     THE START ADDRESS OF THE DATA AREA CONTAIN-
                ING THE IMPULSE RESPONSE.
        IN ADDITION A COMMON DATA BLOCK LABELLED "PARAM" IS
        REQUIRED. WHICH CONSISTS OF THREE WORDS. THE FIRST
        TWO OF WHICH DEFINE THE START ADDRESS-1 OF THE CYCLIC
        BUFFER AND ITS SIZE. THE THIRD LOCATION IS FOR THE
        PASSING OF DATA.
$
$LOWER COMMON/PARAM
```

```
CBPA/3
$PROGRAM
$ENTRY
FILT-   LDA ,0          (CLEAR THE DOUBLE LENGTH RESULT.
        STO *111
        STO *112
        LDA /1+         (FETCH THE NUMBER OF COEFFICIENTS IN
        NEQ ,*177777    (THE IMPULSE RESPONSE AND TAKE THE 2'S
        ADD ,*1         (COMPLEMENT IN ORDER TO MAKE A COUNT.
        STO *113
        LDA /1+         (FETCH THE START ADDRESS-1 OF THE CO-
        STO *115        (EFFICIENTS INTO A POINTER.

MPLY-   CAL .OPCB       (FETCH DATA FROM CYCLIC BUFFER.
        LDA CBPA+2
        STO *116        (SET UP DATA FOR MULTIPLICATION.
        LDA /*115+      (FETCH APPROPRIATE COEFFICIENT.
        STO *117
        LDA ,*115
        STO 1
        CAL .ZMULT
        CLR M
        LDA *121
        ADS *112
        SET M
        LDA *111
        ADD *120
        CLR M
        STO *111
        ICZ *113 MPLY   (FINISHED ?
        RTC
$END


$SEGMENT SCALEFACTOR
$PROGRAM
$ENTRY
SFAC-   LDA /1+
        STO *121
        LDA /1+
        STO *122
        LDA ,0
        STO *123
        CMP *121
        JBC Z REPT
        CMP *122
        JBS Z OUT
REPT-   JBS 15 *121 PAST
        JBS 14 *121 OUT
        JMP .CONT
PAST-   JBC 14 *121 OUT
CONT-   LDA *121
        SET M
        SLA 1 *122
        CLR M
        STO *121
        LDA ,1
        ADS *123
        JMP .REPT
OUT-    LDA *122
```

```
        STO /1-
        LDA *121
        STO /1-
        RTC
$END
$FINISH
```