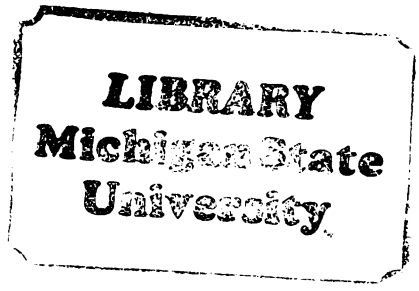

This is to certify that the

dissertation entitled

A MODULAR DISTRIBUTED MICROCOMPUTER SYSTEM
FOR SCIENTIFIC INSTRUMENTATION

presented by

Bruce Hewitt Newcome

has been accepted towards fulfillment
of the requirements for

Ph.D. degree in Chemistry

C G Enke
Major professor

Date Feb. 5, 1984

A MODULAR DISTRIBUTED MICROCOMPUTER SYSTEM FOR SCIENTIFIC INSTRUMENTATION

by

Bruce Hewitt Newcome

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirments
for the degree of

DOCTOR OF PHILOSOPHY

Department of Chemistry

1983

ABSTRACT

A MODULAR DISTRIBUTED MICROCOMPUTER SYSTEM
FOR SCIENTIFIC INSTRUMENTATION

by

Bruce Hewitt Newcome

A hierircal computer system has been designed to meet the needs of chemical instrumentation. This system consists of a central minicomputer that is connected to several dedicated microcomputer systems. These microprocessor systems perform instrument control and data acquisition and then transfer the data to the host system for further data analysis. This separation of tasks allows each computer system to be specifically tailored to the tasks that it performs.

A modular microprocessor system was designed to permit the easy implementation and adaptation of the computer system to the needs of the instrument. A system is assembled from a set of standard single function modules which provided the needed functions for the control system by mounting them on a dual-bus board. This board provides the local bus to interconnect the different modules in the system. Examples of the systems that have been implemented are a laser lab data acquisition system, a diode array

spectrophotometer system, and a time dispersed magnetic sector mass spectrometer.

A distributed processing system was developed for instrumentation problems that require more processing power than a single microprocessor can provide. The use of several distributed microprocessors instead of a single larger processor has advantages in the areas of speed, non-interference, and flexibility. Dedicated interprocessor hardware was designed to implement the different modes of interprocessor communication. These modes of communication are block data transfer, task assignment, parameter transfer, and task coordination.

A four processor system was built to control a Triple Quadrupole Mass Spectrometer. The four processors in the system are a master processor, an ion path processor, a detection processor , and a reduction processor. The master processor handles the mass storage, communications with the operator, and controls the other three slave processors. The ion path processor controls the potentials of all of the elements in the ion path. The detection processor performs the data acquisition and the reduction processor performs peak finding in real time on the acquired data. This system is a significant improvement over a similar single processor system.

To that old curse

" May you live in interesting times."

ACKNOWLEDGMENTS

I would like to thank Professor Christie Enke for his help and guidance. I am also indebted to him for the support and freedom that he provided for this work. I also thank Professor Stanley Crouch and the other members of my committee, Professor George Leroi and Professor David Fisher, for their contributions. I am also indebted to the many members of the Enke and Crouch groups for their friendship and assistance. The financial support of the Office of Naval Research and Extranuclear, inc. is gratefully acknowledged.

I would like to thank my parents for their guidance and support and Professor H. V. Malmstadt and Professor Jim Avery for their initial assistance and direction. I would like to acknowledge the help of the many members of the Chemistry Department Staff that assisted in this work.

Finally I acknowledge the help and friendship of my Co-conspirator Carl Myerholtz without whom this would not have been possible. For truly the whole is greater than the sum of the parts.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1 -INTRODUCTION-

COMPUTER IN CHEMICAL ANALYSES

To answer the question of how computers can be used to help the analytical chemist it is useful to examine the general processes that occur during a typical chemical analysis. An analysis can be broken down into four general stages: sample preparation, interaction between the sample and a transducer(s), recording the output of the transducer(s) as a function of other controlled or measured variables, and processing and interpretation of the data.

SAMPLE PREPARATION

The sample preparation stage involves operations which transform the raw sample into a state where it can effectively interact with the transducer and the elimination of substances in the raw sample that would interfere with the interaction of the transducer and the

component of interest. Examples of these operations include extraction, dissolution, vaporization, chemical reaction, irradiation, and gas and liquid chromatography. A computer can be used to assist in several ways with these operations. For routine analyses computer controlled robots could be used to completely automate the sample handling involved in sample preparation. Computer control of GC and LC equipment greatly increases the ease with which temperature and solvent programing can be accomplished. A less obvious way in which computers can simplify the task of sample preparation is that they enable multiple dimensions of information about the sample to be collected. When this information is cross-correlated it permits the desired information about the component of interest to be determined even in the presence of many other components. This added selectivity allows a much less pure sample to be analyzed thus decreasing the amount of sample purification that must be performed(1).

INTERACTION BETWEEN THE SAMPLE AND TRANSDUCER

The second stage of an analysis is the interaction between the sample and a transducer. The transducer can take many forms such as electric and/or magnetic fields, the various types of electromagnetic radiation, or chemical

reactions. It can also be either a single step process, such as the interaction of the sample and electrode in an electochemical analysis or multiple step process, such as a mass spectrometer where the sample is ionized, interacts with an electric and/or magnetic field, and finally is detected by an electron multiplier. One way that computers can assist in this stage of the analysis is by allowing the use of highly nonlinear transducers. These transducers are difficult to employ without computer assisted correction of the transducer's response curve. Another possible way that computers can assist in the interaction of the sample and the transducer is that the computer can vary the conditions of the interaction in a rapid and intelligent fashion to maximize the "real" information that is acquired from the sample while minimizing the amount of sample required. This can result in increased precision and sensitivity in the analysis.

RECORDING THE OUTPUT OF THE TRANSDUCER

Recording the results of the interaction between the sample and the transducer is the third stage of an analysis. The use of a computer for recording the transducer's output has advantages over conventional techniques in a number of different situations. One of

these situations occurs when data are generated by the transducer at high rates. A modern computer system can record data at rates that are much higher that manual techniques. Another situation happens when large amounts of data are generated by the experiment. Computer systems can be very useful in this case because they allow the data from the experiment to be processed in real time, thus greatly reducing the amount of data that must be stored. A final example of a situation where computer acquisition of data is an advantage is when an experiment takes a long time to perform. Unlike most human beings, a computer can take data for days at a time without ever getting bored. Also computers can simultaneously take related or corrective data, such as temperature for a temperature-sensitive transducer or data from multiple transducers with different selectivity coefficients(2). The computer is also capable of evaluating the quality of data in real time and thus move to the next point when the desired signal-to-noise ratio is reached or if the present datum is not of interest. A further advantage of computer data acquisition systems is that data are recorded in a machine readable form. This allows the computer to assist directly in the further reduction and processing of the data.

PROCESSING AND REDUCTION OF DATA

A stage of the analysis where computers can have a large impact is the reduction and analysis of the raw data from the experiment. An example of analytical techniques where the use of computers have had a large impact are Fourier transform NMR and IR. The routine use of these techniques depends entirely on the use of computer systems to acquire data and to perform the Fourier transform operation. Other examples of computer assisted analysis of data are the integration of peak area for a number of instrumental methods, the deconvolution of overlapped peaks, and the use of computer searched libraries of standard compounds. A future area that has a great deal of potential is the use of artificial intelligence systems for both data analysis and experiment definitions. These experiment definitions would provide rules for intelligent real time optimization of the experimental parameters.

HISTORICAL PERSPECTIVE

From an historical perspective the widespread use of computers for chemical instrumentation began with the

advent of the minicomputer. These were the first computers that could operate in the laboratory environment and which were directly accessible to the chemist. Minicomputers were, however, still much more expensive than many of the instruments in the laboratory. Thus, the use of a dedicated minicomputer was only justified with large expensive instruments, such as NMR or mass spectrometers, with high data rates. In an attempt to share the cost of the computer among several projects these minicomputer systems were frequently connected to a variety of different instruments(3)(4)(5). Because of the complex and expensive interfaces between the computer and the instruments, the computer frequently had only minor control over the instrument and was normally used just for data acquisition and analysis. Even with these limitations the use of minicomputers greatly increased the quantity and quality of information that could be obtained from an instrument.

The next major development in the use of computers for chemical instrumentation was the development of the microprocessor. The low cost of these processors has led to an explosion in the use of computers to control chemical instruments. The use of microprocessors has become so prevalent that most new chemical instruments either come with built-in computers or have them available as options. These systems are usually built from either commercial

microprocessor boards or are fully custom systems. These approaches can be justified because of the needs of commercial production. In the research laboratory where existing or custom instruments must be interfaced to computer control systems, several different approaches have been employed(6),(7). The simplest of these approaches is the use of "Personal Computers" to control chemical instruments(8),(9)(10). This method is attractive when only a few experimental parameters need to be controlled and the resulting data rates and data volumes are small to moderate. The personal computer contains most of the needed components for a control system so only the interfaces to the instrument must be designed. The disadvantage of this approach is the lack of flexibility of the system and the cost of any unused features that are included as part of the basic computer.

A second approach is the use of standard industrial computer boards(11),(12) for instrument control systems. This is a more flexible approach than the use of personal computers since a wide variety of different functions are available from different manufacturers. However, many of the available boards contain more than one function due to the fixed board size, which sometimes results in the purchase of unwanted capabilities. A third approach that has been employed in research laboratories that are

interested in laboratory automation is the use of microcomputer systems that have been specially designed for laboratory control systems(13),(14). This approach offers the greatest flexibility in the configuration of the microprocessor system but requires a large investment of time and personnel to design the needed components. This investment of time is offset by the generation of efficient and economical microprocessor systems that are readily tailored to the specific needs of the instrument.

SEPARATION INTO ANALYSIS AND CONTROL SYSTEMS

From the initial discussion of the use of computers to aid chemical analyses, it can be seen that there are two general areas where computers can be of assistance. These two areas are real time instrument control and data acquisition, and the interactive analysis and reduction of data that result from an experiment. The two different classes of functions have different requirements from the computer system in terms of response time to external events and in the types of peripherals needed. The real time control applications require a rapid response time from the processor and need dedicated interfaces to the

instrument being controlled. The data analysis functions on the other hand require only a moderate response time and utilize peripherals which are capable of storing large amounts of data (e.g. disk and tape drives)or that are useful for presenting data to the operator (e.g. graphics displays, printers and plotters). These differences in requirements suggest that two different computers connected in a hierarchical configuration might best meet the needs of both types of functions. This would allow each of the computers to be tailored to the different requirements of each class of functions.

Examples of data analysis functions are reference library searching, correction of the raw data for background effects, transform operations that change the raw data into a more comprehensible form, signal-to-noise enhancement, correlation analysis, interactive graphical analysis of the data, and presentation and tabulation of the experimental results. A minicomputer or one of the new super-microcomputers have several features that are helpful in implementing these data analysis functions. These computers are capable of utilizing large amounts of memory, can be readily interfaced to large disk and tape drives, and can implement multi-user operating systems efficiently. This last feature allows the cost and capabilities of expensive peripherals to be shared by several users. These

computer systems are also capable of supporting specialized numeric and array processors which can greatly increase their number crunching ability. Interfaces for graphics displays and plotters are also readily available. Figure 1-1 shows a typical example of this type of computer that is used as a data analysis system for several different instruments. One final advantage of separating the control and data analysis functions into different processors is that the control system can be used to collect data for a new experiment while another user is analyzing the data from a previous experiment on the data analysis system.

Some examples of the types of functions that a control system can perform are the control of instrument parameters, the acquisition of data from the instrument, real time data reduction to eliminate extraneous data values, and rudimentary displays of the acquired data for use in optimization of the instrument parameters and monitoring the course of an experiment. To perform these functions the control system needs a variety of interfaces to the instrument and peripherals capable of supporting a "user friendly" interface with the operator. Modern microprocessors have several features which make them ideally suited for implementing instrument control system. One feature is high performance at a low cost which allows the microcomputer to be dedicated to a single instrument.

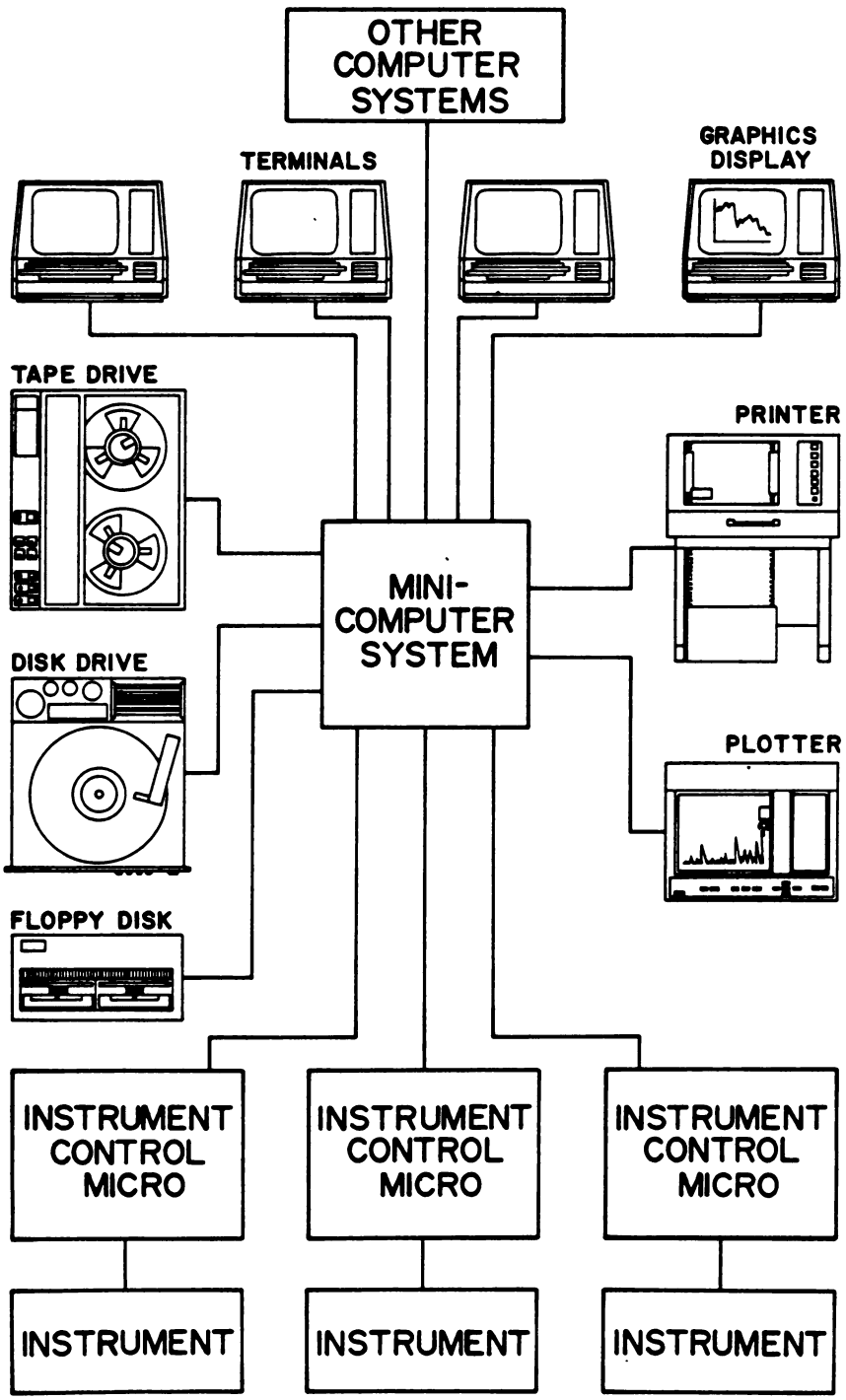

Figure 1-1 A Typical Hierircal Computer System.

Another feature that these microprocessors have is a less complex bus structure than the minicomputers which allow simpler interfaces between the instrument and the computer to be constructed. The availability of LSI peripheral circuits, such as USARTs, parallel I/O ports, and microprocessor compatible signal converters (ADCs and DACs), also simplifies the design of new interfaces to the instrument. The programmable nature of many of these LSI peripherals is another advantage because it allows a general purpose interface to be used for a variety of instruments by merely changing the controlling software. The low cost of these peripherals also increases the number of instrument parameters that can be controlled by the computer at a reasonable cost.

COMPUTER NETWORKS

The advent of separate data analysis systems and instrument control systems has led to the development of a variety of laboratory networks which allow the different instrument control systems to communicate with other "host" computer systems(15),(16). These host computer systems provide the "data analysis" system functions while the

satellite systems provide individual instrument control and data acquisition. These heriarchical systems permit the different functions that are involved in an experiment to be performed on the computer that is most suitable for that task. The existence of these networks also assists in the combination of data from different types of instruments to obtain the answers to a single analytical problem. Modern control systems are developing in several directions such as intelligent systems(17) which employ the results of an experiment to determine the conditions for the next experiment, and the use of distributed processing systems to provide increased capabilities for instrument control.

LABORATORY MICROCOMPUTER SYSTEM DESIGN GOALS

To meet the needs of our research group for instrument control systems, we decided to design a microcomputer system that would better meet the requirements of a research environment than commercial microprocessor boards or personal computers. This system was designed using experience gained with a microprocessor system designed by Jim Avery(18) and Dan Lovse(19)at the University of Illinois , and a previous microprocessor system designed at

Michigan State University by Erik Carlson(20) and Jim Hornshuh(21). To insure that the system would meet the needs of the research environment, a list of design goals was compiled. These goals are given in Table 1-1 and discussed below.

The requirement for the hardware to be modular on the single function level arises from the constantly changing nature of most research instruments. This modularity allows the control system to be easily modified as the experimental needs change. The single function nature of the modules permits the addition of just the capabilities that are required without wasting time, money, or board space on functions that are not needed. The design goal of operation with a wide range of configurations arises from the vastly different requirements of different instruments in a research laboratory. Another of the design goals is the need for the computer systems to be able to operate in a hierarchical system with other computers. This allows the microcomputer to be used as a control system with a separate minicomputer acting as a data analysis system.

The microcomputers also need to be inexpensive if they are to be dedicated to a large number of different instruments. The availability of graduate student labor permits the assembly of the microcomputers to be labor

Table 1-1. The Design Goals for the Microprocessor System.

1. Modular hardware on a single function level.

2. Capable of operating over a wide range of configurations.

3. Should be able to operate in a hierarchical environment.

4. Inexpensive given low cost labor.

5. Elimination of card edge connectors for improved reliability.

6. Provisions for I/O connections on the backplane.

7. Should be able to evolve gracefully from a single processor into a distributed processor system.

intensive and still result in an affordable microcomputer. To improve the reliability of the system, the elimination of printed circuit board edge connectors was a design goal. To be reliable these connectors require gold plated fingers on the PC board and accurate PC board routing. Attempts to use edge connectors had resulted in problems in previous systems since neither of these requirements could be easily met with the available in-house equipment. Another design goal that resulted from experience with previous systems was the desire that I/O connections would be made through the system backplane. This allows a board in the system to be easily removed or extended for debugging without dragging a mass of cable with it. A final design goal was the requirement that a single microprocessor system should be upgradable to a distributed processing system without complete redesign of the instrument interfaces.

SUMMARY

This thesis describes a microcomputer system which meets these goals. The single processor implementation is described in Chapter 2 and several of the various applications to which it has been put are described in

Chapter 3. The concept of distributed processing and the advantages that it affords are discussed in chapter 4. Chapter 5 describes the specific hardware that was designed to implement a multiple processor version of this microcomputer system. This multiple processor system was used to implement a control system for an advanced mass spectrometer as described in Chapter 6. A short summary of the software that was written for these systems is presented in Chapter 7. This microcomputer system has been very successful in our department where it has been used to implement more than twelve different instrument control systems.

CHAPTER 2 - THE SINGLE MICROCOMPUTER SYSTEM -

To meet the design goals stated in Chapter 1 a modular microprocessor system was developed. This system is composed of a series of single function modules which are interconnected to form a computer system by mounting the modules on a dual-bus board. The dual-bus board has two sets of bus traces on it which connect the local processor bus to all of the modules in the system. Up to four of the dual-bus boards can be plugged into a backplane to form a complete microprocessor system as shown in Figure 2-1. The two sets of bus traces on the dual-bus board can be seen in Figure 2-2, which shows the foil side of a dual-bus board. The signal traces on the dual-bus board are connected to the backplane by pin and socket connectors which mount in the holes shown on the left side of Figure 2-2. These connectors were used because of their high reliability compared to conventional edge connectors. As can be seen in Figure 2-2, the lower set of bus traces is directly connected to the pin connector, while the top set of traces is left unattached. The uncommitted nature of the top set of bus traces provides a great deal of flexibility in designing different microprocessor system.

USES OF THE SECOND BUS

On most of the dual-bus boards in a single processor system, the two sets of bus traces are joined by a series of vertical jumpers so that both sets of bus traces are connected to the local processor bus. This allows standard function modules to be mounted on both the top and bottom halves of the dual-bus board, which increases the number of modules that can be mounted on a single dual-bus board. However, the top bus can be used for more novel applications by jumper-connecting the traces to the backplane instead of to the lower bus. This allows the top bus to be used as an interprocessor bus, a hardware controlled peripheral bus, or as the local processor bus of another processor. The use of the top bus as an interprocessor bus is discussed in the second half of this dissertation. The advantages of employing the top bus as a peripheral bus are examined in chapter 3 where a linear diode array spectrophotometer control system is shown as an example. An example of the use of the top bus as the local bus of another processor would be the implementation of an intelligent data base controller. A module that connects to both of the busses would allow requests for information to be transferred to the data base control processor, which

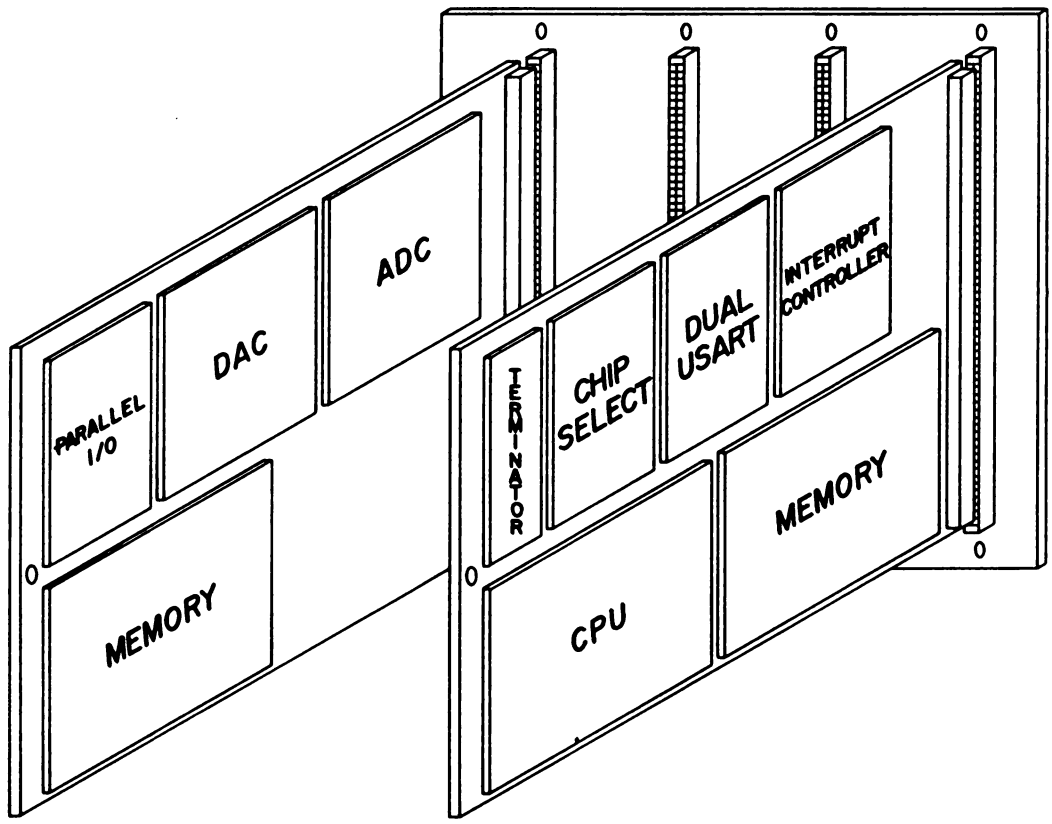

Figure 2-1 A Typical Single Microprocessor System.

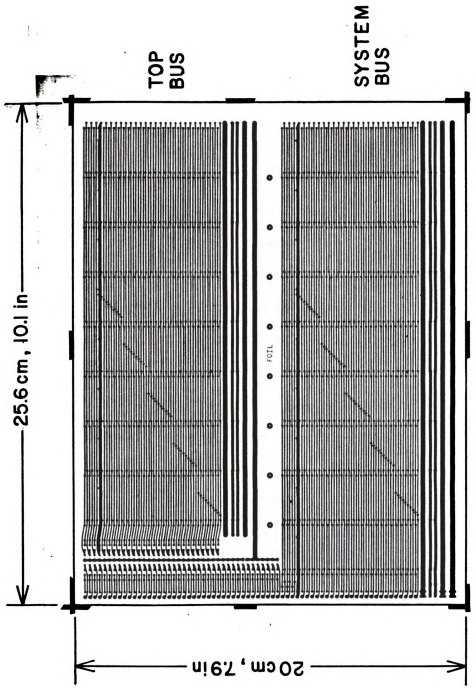

Figure 2-2 The Foil Side of the Dual-Bus Board.

would then search the data base for the requested information and return it to the main processor.

THE BACKPLANE

The backplane board, which is shown in Figure 2-3, is capable of interconnecting up to four dual-bus boards. The backplane has a set of traces along it's lower edge which connects the local bus between the dual-bus boards and also distributes power. The signals from the connectors on the top half of the backplane are brought out to sets of pads which permits the mounting of different types of cable connectors. The latter are used to connect the computer system to the instrument that is being controlled and to the rest of the real world. Thus, the dual-bus boards can be removed or extended without having to disconnect any cables. Sets of pads are also provided for connecting the local bus to another backplane if more than four dual-bus boards are required to construct the system.

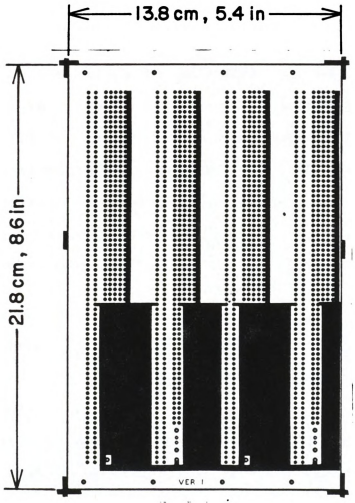

Figure 2-3 The Trace Side of the Backplane Board.

LOCAL BUS DEFINITION

The local processor bus used in this system is a fully demultiplexed synchronous bus. The data and address signals are all active high while most of the control signals are active low to reduce problems with noise spikes. Further information on the signals generated by the 8085 and 8088 processors is found in the appropriate Intel manuals(22),(23). In addition to the normal bus signals, four additional signals have been provided for the use of the interprocessor modules. The layout of the local processor bus is given in Figure 2-4. The definitions of the signals that make up the bus are as follows:

D0-D7 Eight bidirectional data lines.

A0-A15 The lower sixteen address lines.

A16-A19 The four most significant address lines (8088 system only)

ALE The falling edge of Address Latch Enable indicates that a stable address is available to be latched into the address latches. It is also used to indicate the start of a bus cycle.

WR\ The active LO write signal is used to strobe data from the data bus into the addressed location.

		*	Shield Gnd
SHIELD GND	*		
		*	HALT (A19)
(A18) PS1	*		
		*	PS2 (A17)
(A16) PS3	*		
		*	PHOLD\
PHLDA	*		
		*	PWR\
PRD\	*		
		*	HOLD\
HLDA	*		
		*	RESIN\
READY	*		
		*	RST7.5 (INT0)
(INT1) RST6.5	*		
		*	RST5.5 (TEST\)
INTR	*		
		*	RESET\
CLK\	*		
		*	INTA\
IO/M	*		
		*	S1
RD\	*		
		*	WR\
ALE	*		
		*	A15
A14	*		
		*	A13
A12	*		
		*	A11
A10	*		
		*	A9
A8	*		
		*	A0
A1	*		
		*	A2
A3	*		
		*	A4
A5	*		
		*	A6
A7	*		
		*	D0
D1	*		
		*	D2
D3	*		
		*	D4
D5	*		
		*	D6
D7	*		
ANALOG GND	*		
-24	*		
+24	*		
+5	*		
DIGITAL GND	*		

Figure 2-4 The Layout of the Local Bus.

RD\ The read signal is used to enable a memory or peripheral device to drive the data lines. This signal is active LO.

S1 This signal gives an advanced indication of the direction of information transfer on the data lines. A HI level indicates data are to be read into the processor.

IO/M A HI level on this signal indicates an I/O transfer is to take place while a LO level indicates a memory transfer is to occur.

INTA\ An active LO interrupt acknowledge from the processor.

CLK\ The inverted processor clock.

RESET An active LO reset signal from the processor which is used to initiallize all of the peripherals in the system.

INTR An active HI general interrupt input to the processor. It is normally generated by an 8259A interrupt controller.

RST5,5-RST7,5 Three interrupt signals to the 8085 processor which have predefined interrupt vector locations in memory.

INT0,INT1 Two bussed inputs to an 8259A interrupt controller in an 8088 system.

TEST\ An active LO signal that is tested by the 8088 CPU

during the WAIT instruction.

READY When this signal is LO the processor enters a wait state until the ready signal returns HI. It is used to slow down the bus cycle for slow peripherals.

RESIN\ An active LO reset input to the processor.

HLDA A hold acknowledge signal which indicates that the processor has released the local bus when HI.

HOLD\ An active LO signal which requests the use of the local bus from the processor.

PRD\,PWR\,PHOLD\,PHLDA Four signals used by the interprocessor modules to control the generation of the RD\, WR\, HOLD\, and HLDA signal on systems which have interprocessor modules.

PS1-PS3 Peripheral status lines which are used by the 8085 CPU module and the interprocessor modules.

HALT\ An indication that the 8085 CPU has executed a halt instruction.

STANDARD MODULES

More than thirty different standard modules have been designed for use in this system. This wide range of functions greatly reduces the number of custom interfaces

that must be designed when a new instrument control system is constructed. Table 2-1 gives a list of the currently available modules. These modules are constructed in two different heights, single height modules that are 3.7 inches high and double height modules which are 7.7 inches high. The modules come in different widths depending on the amount of space that the function needs. The modules can be designed in one inch increments from a minimum of one inch to a maximum of nine inches. This modularity of size combined with the ability to pick up bus signals at multiple locations on the module result in very efficient use of space.

PROCESSOR MODULES

There are two different processor modules; one uses the Intel 8085 processor and the other uses the Intel 8088 processor. The 8085 CPU module has latches on it which demultiplex the data-address signals from the processor and tri-state drivers which buffer the data and control signal. This module also contains two switches which allow the processor to be reset and held. There are six LEDs on this module which provide indicators for the following signals, PS1,PS2,PS3,HLDA,HALT,and SOD (serial data out from the 8085). The 8088 processor module combines the functions of

Table 2-1. A list of Standard Modules.

8085 CPU	9513 CTC *
8088 CPU	58167A RTC
RAM/ROM	Octal 12 bit DAC (AD390)
8K RAM	Dual 8-bit DAC (AD558) *
8259A Interrupt Controller	Dual 12-bit DAC (1230) *
Chip Select	12-bit ADC (AD574) *
Address Extender	Diff. Multiplexer *
Wait State Generator	Programable Gain Amp. *
AC Terminator	Diff. Transceiver
Active Terminator	Softknobs Interface
SCSI Interface	8087 Adapter
1771 Floppy Disk Controller *	2",3" Wire Wrap
Dual 8251A USART	Bus Multiplexer
8255A PIO	Graphics Controller (7220)
8253-5 CTC	Graphics Memory Plane

* Modules that were designed by other people in the department.

the processor with an interrupt controller, chip select logic, and an active terminator on a single double height module. It was necessary to combine these functions to provide enough space for the processor functions in the same area as an 8085 CPU module, an interrupt controller module, a chip select module, and an active terminator module. In addition to circuits which demultiplex the data and address signals and buffer the data and control signals, the 8088 CPU module provides circuits which control the restart address of the processor and provide for the addition of zero, one, or two wait states for memory locations between XF000 and XFFFF. This wait state circuitry is necessary for the processor to access the standard LSI peripheral modules. The functions of the interrupt controller , the chip select logic, and the active terminator are the same as those provided by the standard modules. An adapter module has also been designed which allows the 8087 numeric coprocessor to be used with the 8088 CPU module.

MEMORY MODULES

There are two different memory modules that can be used with this system. The first provides up to 8 Kbytes of RAM memory using 2114 (4 x 1K bit) memories. This module

has been largely replaced in recent systems by the second memory module. This module, called the RAM/ROM module, has sockets for eight 24-pin JEDEC standard RAM or ROM chips. It can provide a total of 8 KBytes of memory if 1 or 2 Kbyte chips are used or 16 Kbytes if 2 or 4 Kbyte memory chips are used.

SYSTEM SUPPORT MODULES

Several modules have been designed to provide system support functions such as interrupt control, chip select logic, wait state generation and bus termination. The interrupt control is provided by a module which employs an Intel 8259A interrupt controller to provide eight vectored interrupts. This module can be used with both the 8085 and 8088 CPU modules. Flip/flops and inverters are provided on this module for use in transforming interrupt signals into the active HI level interrupt signals that the interrupt controller requires.

The address decoding for all the peripheral modules is performed by a separate chip select module. This module decodes 512 addresses in memory space into eight 64 byte unqualified chip selects, and the top unqualified chip select is decoded into eight 8 byte read/write qualified

chip selects. The unqualified chip selects are normally used by other LSI peripheral modules in the system while the qualified chip selects are used to control flip/flops and latches. By combining the address decoding onto a single module, the other modules are simplified and duplication of the chip select logic is avoided. To allow the entire 1 megabyte address space of the 8088 to be used, an address extender module was designed which decodes address lines A16 through A19. The address extender module may be used with all the chip select and memory modules.

The wait state module is used in an 8085 system to stretch the bus cycle by driving READY LO. Its function is provided on the 8088 CPU module because most of the LSI peripherals require a longer bus cycle than the 8088 normally provides. Bus termination is provided by two modules, an active terminator module and an AC terminator. The active terminator module terminates the bus signal with a 1 kilohm resistor to a 2.6 volt source which is formed from a zener diode and an emitter follower. The 1 kilohm resistance was chosen as a compromise between the need to match the characteristic impedance of the bus traces (approximately 200 ohms) and the desire to keep the required DC current levels as low as possible. The AC terminator was designed to permit additional termination to be added to a system without increasing the level of DC

drive required. It terminates the bus lines with a 220 ohm resistor and a 10,000 pF capacitor in series. This combination matches the characteristic impedence of the bus traces at the frequencies that the processor uses.

MASS STORAGE INTERFACE MODULES

Interfaces to mass storage devices are provided by two modules: a floppy disk controller and a SCSI(Small Computer Standard Interface) adapter module. The floppy disk controller was designed using a 1771 floppy disk controller chip and is capable of controlling up to four 8" floppy disk drives. This module was designed by Ralph Thiim(24). The SCSI adapter module can be used to interface the computer to a variety of intelligent disk and tape drive controllers using the industrial standard SCSI bus structure. The use of the SCSI controller greatly simplifies the software since it allows a single driver program to control both floppy disk and Winchester disks.

LSI PERIPHERAL MODULES

Serial ports are provided by a dual USART module that has two 8251A USARTs. The baud rates are set by jumpers

which control a baud rate generator chip. Two USARTs were used so that the baud rate generator and the line driver chips could be shared. A Parallel I/O module using an 8255A PIO chip provides parallel input and output ports for the system. Two different counter/timer modules have been designed for this system. One uses the Intel 8253-5 counter/timer chip while the other uses the Advanced Micro Devices 9513 counter chip. These two circuits differ in their complexity and capabilities. The 8253-5 is the simpler of the two and has 3 sixteen-bit counters and fewer operating modes than the 9513. The 9513, which has 5 sixteen-bit counters, is much more flexible than the 8253-5 and includes such features as an on-chip frequency source and a wide variety of operating modes. The 9513 module was designed by Gene Ratzlaff(25). A real time clock module has been designed to provide the computer with the time of day. This module, which uses the National 58167A clock/calendar chip provides both time of day and date for the processor.

ANALOG FUNCTION MODULES

Analog interface functions are provided by a number of different modules. These include three different DAC modules, an ADC module, a programmable gain amplifier, and

a differential multiplexer module. The simplest of the DAC modules is a dual 8-bit DAC module which contains two AD558 unipolar DACS. The second DAC module is an octal 12-bit DAC module which uses two AD390 quad DAC chips. This module provides eight 2's complement bipolar DACs. The third module is the most flexible of the three modules. This module can be configured to use either 8 or 12 bit DACs; either DAC can be independently configured as unipolar or bipolar with a full-scale range of 10.0, 7.5, 5.0, or 2.5 volts. The ADC module uses an AD574 ADC which is a 12-bit 35 microsecond converter. This module also contains a AD583 sample and hold circuit. The differential multiplexer was constructed with a HI1-0507 which is an eight channel differential multiplexer. A Burr-Brown 3606B6 was used to implement the programable gain amplifier module. This module is software programable for gains between 1 and 1024 in powers of 2. Most of these modules were designed by other people in the department: the differential multiplexer and programable gain amplifier were designed by Lynn Jones(26), the ADC was designed by Gene Ratzlaff(25), the AD558 module was designed by John Stanley(27), and the dual 8/12 bit DAC board was designed by Norm Penix(28).

MISC. MODULES

To aid in the construction of custom interfaces several miscellaneous modules have been designed. A differential transceiver module was designed for use when the interfaces to the instrument needed to be physically distant from the computer system. This module allows a subset of the local bus to be differentially driven to a remote location. The ability to place interfaces close to the instrument results in much shorter analog signal paths which leads to better accuracy and greater noise immunity. For use in prototyping interfaces, two wire wrap boards have been designed. These modules come in 2 inch and 3 inch widths with pads provided on the 2 inch module for connections to the local bus. A final miscellaneous module is a softknob interface which can interface up to four quadrature encoded rotary knobs to the computer system. These knobs are very useful in providing a convenient way of controlling various instrument parameters.

DUAL HEIGHT MODULES

Several modules have been designed which make special use of the top bus. The first of these is a bus exchanger

module which can switch the two sets of bus traces on a dual-bus board between the local bus and a second bus connected to the top of the backplane. This module is normally used to switch blocks of memory between two different busses. Two other modules use the top bus as a graphics memory bus. These modules are the graphics controller and the graphics memory plane module. The graphics controller uses a NEC7720 graphics controller chip. This chip controls up to 256K bytes of private graphics memory and can draw points, lines, boxes, and circles. The graphics memory module can use either 16K or 64K DRAM chips which results in a 512 by 512 pixel display if 16K chips are used or a 1024 by 1024 memory plane if 64K memory chips are used.

SUMMARY

This microcomputer system has been used to construct more than twelve systems in our department with excellent results. These systems have been very reliable with no development of contact problems that plagued previous microcomputer systems in the laboratory environment. The modular nature of this system has also allowed systems with

very different functions to be constructed with only minor amounts of custom interface design. Three of the systems that have been built with these modules will be examined in Chapter 3 to illustrate the power and flexibility of this system and the advantages it has for chemical instrumentation.

CHAPTER 3 - EXAMPLE MICROCOMPUTER SYSTEMS -

The modular microprocessor system described in Chapter 2 has been used to construct over twelve different instrument control systems in our department. As examples of the different types of microprocessor systems that can be built, three of these control systems will be examined in detail. These three systems are a data acquisition system for a laser laboratory, a diode array spectrophotometer that is used for optical rotatory dispersion experiments, and a control system for a time-resolved magnetic dispersion mass spectometer. These three systems will also be used to demonstrate several ways the computer was used to improve the chemical information that is obtained from an experiment.

LASER LAB DATA ACQUISITION SYSTEM

The first computer system that will be examined is the laser lab data acquisition system (29). The type of experiment this system was designed to control is the

investigation of saturation effects on absorption spectra. In these experiments, a sample is optically pumped by one wavelength of laser light and then the absorbance at a different wavelength is measured by a probe beam(26). To perform these experiments, four light intensities must be measured for each laser pulse. These four values are the intensity of the incident pump beam, the intensity of the transmitted pump beam, the intensity of the incident probe beam, and the intensity of the transmitted probe beam. A block diagram that shows the major components of this system is presented in Figure 3-1.

The system was constructed on three dual-bus boards using the following standard modules: an 8085 CPU module, two RAM/ROM modules, a dual USART module, an interrupt controller module, an active terminator, a floppy disk controller module, a parallel I/O module, a differential multiplexer module, a programmable gain amplifier module and a 12-bit ADC module. As can be seen in Figure 3-1 one of the USARTs is used to communicate with a terminal while the other USART is used to communicate with a DEC PDP-11/40 which performs advanced data analysis functions. The floppy disk is used to store both system software and acquired data. A "step through" of the stages involved in data acquisition will best illustrate the system operation. A data acquisition operation is started by the firing of the

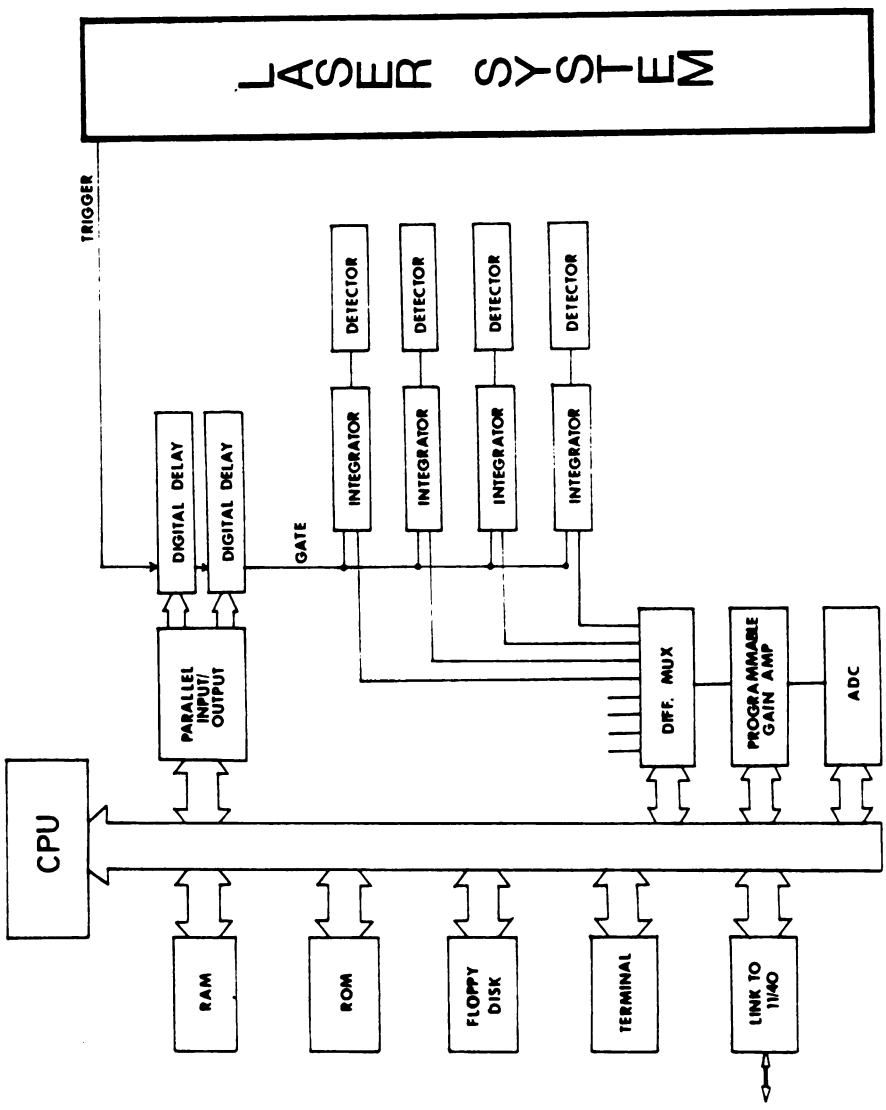

Figure 3-1 Block Diagram of the Laser Lab Computer System.

Nd/YAG laser system, which sends a trigger signal to the microcomputer. The trigger signal is connected to several digital delay cards where it is used to generate gate and reset signals to a set of four gated integrators. The length of time that the gate is delayed from the trigger and the width of the gate pulse are both programmable through the parallel I/O port that controls the digital delays. The four light beams to be measured are converted to electrical current by four photodiodes, and the resulting signals are integrated by four separate gated integrators. The end of the gate pulse causes an interrupt signal to be sent to the microprocessor. The microprocessor then acquires the four data values by selecting each of the four channels of the differential multiplexer sequentially, setting the programmable gain amplifier to the proper gain for the selected channel, and converting the resulting signals. This process is repeated for ten to one hundred pulses and then the data are displayed along with the transmittance for each of the two beams. The software also displays the mean transmittance, standard deviation of the mean, and the percent error.

The laser lab system is an excellent example of how a computer system can greatly increase the amount of chemical information that can be acquired. Before the computer system had been constructed a number of experiments were

attempted using an oscilloscope for readout. Polaroid photographs of the oscilloscope screen provided data storage. It was immediately apparent that this approach had serious shortcomings in several areas. The first of these was the task of aligning the optical components in the system. This task was extremely tedious because after every change in the optical path, the ratio of the incident and transmitted signals had to be determined from the oscilloscope display. With each measurement so tedious, the estimation of the error in the measurements was very difficult. Even greater problems were encountered in trying to determine accurate values for the transmittance of the pump and probe beams. Because the use of an oscilloscope allowed only one diode signal to be monitored at a time the transmittance of the two beams could be determined only from average values of the incident and transmitted beams. This was a problem because the pulse-to-pulse variability of the laser power was quite large (factors of two or more are common). The error in the determination of the average beam intensities was larger than the small changes in probe beam transmittance that the experiment was trying to determine. Using the data system, these problems were easily overcome since the computer could measure all of the signals for each pulse of the laser thus providing a complete transmittance measurement for both beams as well

as an accurate measurement of laser power in each beam. The computer system also calculates the error in the measurements which is of great value in trying to align the optical components.

LINEAR DIODE ARRAY SPECTROMETER SYSTEM

The second control system is a data acquisition system for a linear diode array spectrophotometer(30),(31). This system is used to determine optical rotatory dispersion (ORD) spectra of various compounds. The ORD technique involves the determination of the optical rotation of circularly polarized incident light by optically active compounds as a function of the wavelength of the incident light. The diode array that is used as a detector has 512 diodes arranged in a linear array. In operation, light striking the diodes causes a decrease in the charge that is stored in each diode. After a programmable integration time, the charge on each diode is serially shifted out of the array, and the charge that must be added to return the diode to the original level is measured. This charge is proportional the amount of light that the individual diode received.

As can be seen from the block diagram of this system shown in Figure 3-2, the top bus on several of the dual-bus boards is used as a hardware-driven peripheral bus. The reason that this configuration was chosen is that the diode array generates data at a rate faster than can be accommodated by the processor. Normally the data could be stored in memory using a direct memory access (DMA) controller. However, this requires the use of the local bus which inhibits operation of the processor while the DMA transfer is going on. DMA storage was not a viable option in this case because there is insufficient time between blocks of data for the processor to load the integration time and acquisition parameters for the next block of data. These problems were avoided by using a bus exchanger module to switch two blocks of memory alternately between the local bus and the top bus. This allows the data to be stored in memory over the peripheral bus while the processor is loading the next set of acquisition parameters. After several blocks of data have been stored in the memory, the bus exchanger swaps the banks of memory so that the processor can process the data and store it on disk.

The computer system was constructed on six dual-bus boards using the following standard modules: an 8085 CPU module, two RAM/ROM modules, two RAM modules, a dual USART

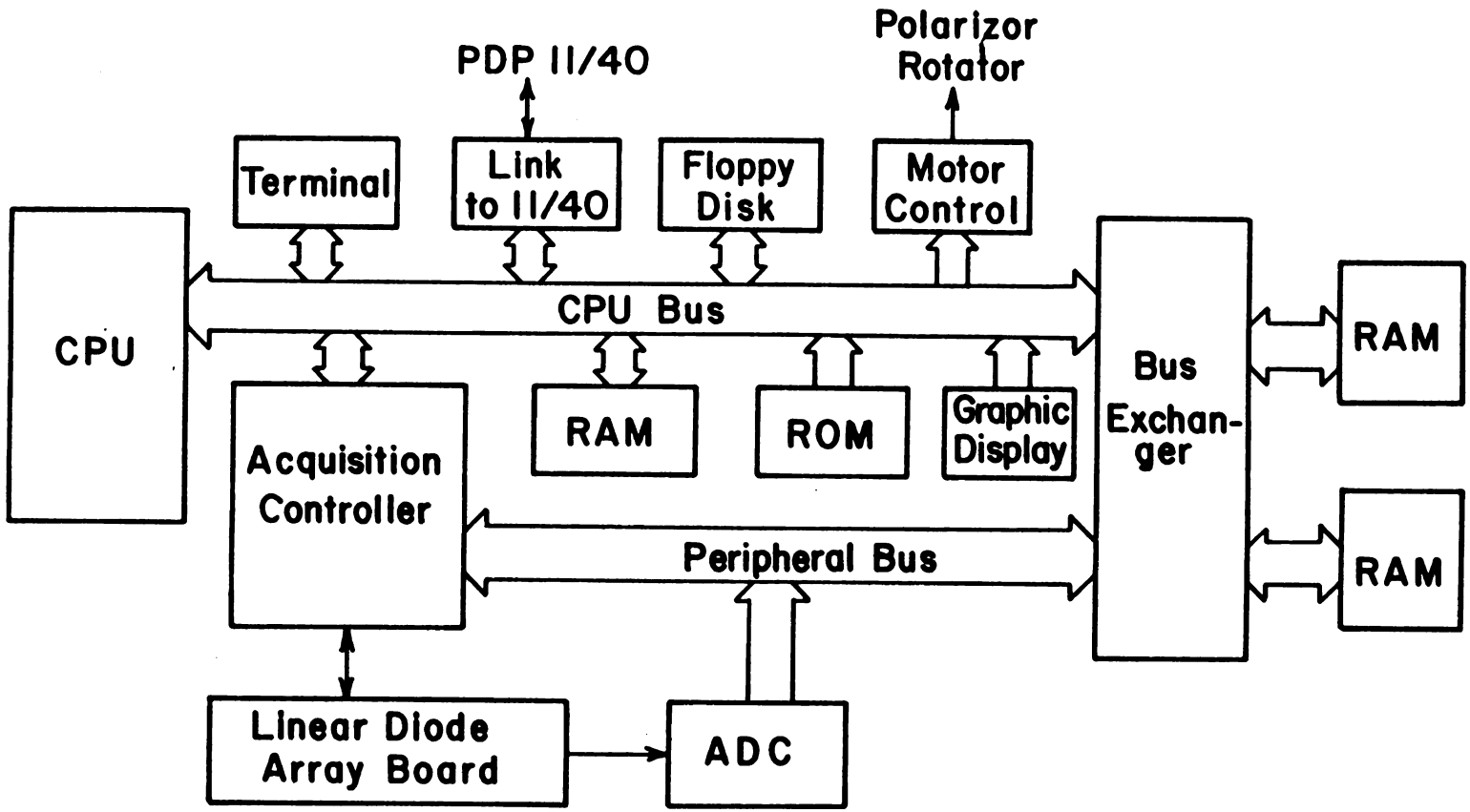

Figure 3-2 Block Diagram of the Diode Array System.

module, an interrupt controller module, a floppy disk controller module, an active terminator module, and a bus exchanger module. In addition to the standard modules, a number of custom modules were designed for stepper motor control, control of the diode array board and the data acquisition hardware, and an interface to a commercial graphics display board. The stepper motor is used to rotate an optical polarizer so that the absorption at different rotations of light can be measured. The graphics display was provided so that the operator could check the data to determine if the instrument is working properly. The data can then be shipped up to the PDP 11/40 for further processing using one of the serial ports.

A unique feature of the acquisition hardware is an "autoranging" function which allows a wide dynamic range of data to be acquired with a moderate amount of memory. With this function, a scan of data is acquired and stored in memory using a short integration time. Then integration time is then doubled, and a new scan of data acquired. However, the individual values are stored in memory only if they are below a preset threshold that is near full-scale for the analog-to-digital converter. By storing these new data points at the same memory addresses as the original data points, and also storing a code which indicates the integration time, the total amount of memory that is

required is only the original 1024 bytes of memory. The integration time is then doubled again and the processes is repeated until the desired dynamic range has been covered. The final result is a block of data that contains a pseudo floating point representation of the intensity at each wavelength. The dynamic range of this autoranging function is limited by the time that the integrations require and the dark current of the diode array. This hardware thus helps to overcome the dynamic range limitations of the diode array detector.

MASS SPECTROMETER CONTROL SYSTEM

The microprocessor system that was built to control the time-resolved magnetic dispersion mass spectrometer is an example of a large single processor system(32). The time-resolved magnetic dispersion mass spectrometer (B-TOF) is a new type of tandem mass spectrometer that was invented at Michigan State University(33). In this mass spectrometer the sample to be analyzed is first ionized in the source region. The ions that are formed are then extracted from the source in a pulse and accelerated to several thousand volts of energy. These ions then enter a

collision chamber where some of the ions break apart due to collisions with neutral molecules. The ions and their fragments are then analyzed by the magnet sector which passes only ions of a given momentum to the detector. If the arrival times of the different ions are measured, it is possible to determine the masses of all the ions, both fragments and precursors, and which precursors are the source of each fragment.

The B-TOF data system, shown in Figure 3-3, was constructed using the following standard modules; an 8088 CPU module with an 8087 numeric coprocessor adapter module, three RAM/ROM modules which provide 8 Kbytes of ROM and 40 Kbytes of RAM, two dual USART modules which provide serial ports for connection to a terminal, a printer, and a PDP 11/44, a real time clock module, an interrupt controller module, a SCSI adapter module which communicates with a SCSI disk controller that is controlling a 5 Megabyte Winchester disk and a floppy disk, a 12-bit ADC module, and two parallel I/O modules. One of the parallel I/O modules is used to control the magnetic field strength through a 16-bit DAC; the other parallel I/O port is used to control a digital delay module and a gated integrator. The digital delay module determines the time delay between the pulsing of the ions out of the source and the measurement of the ion current by the gated integrator. By changing the length

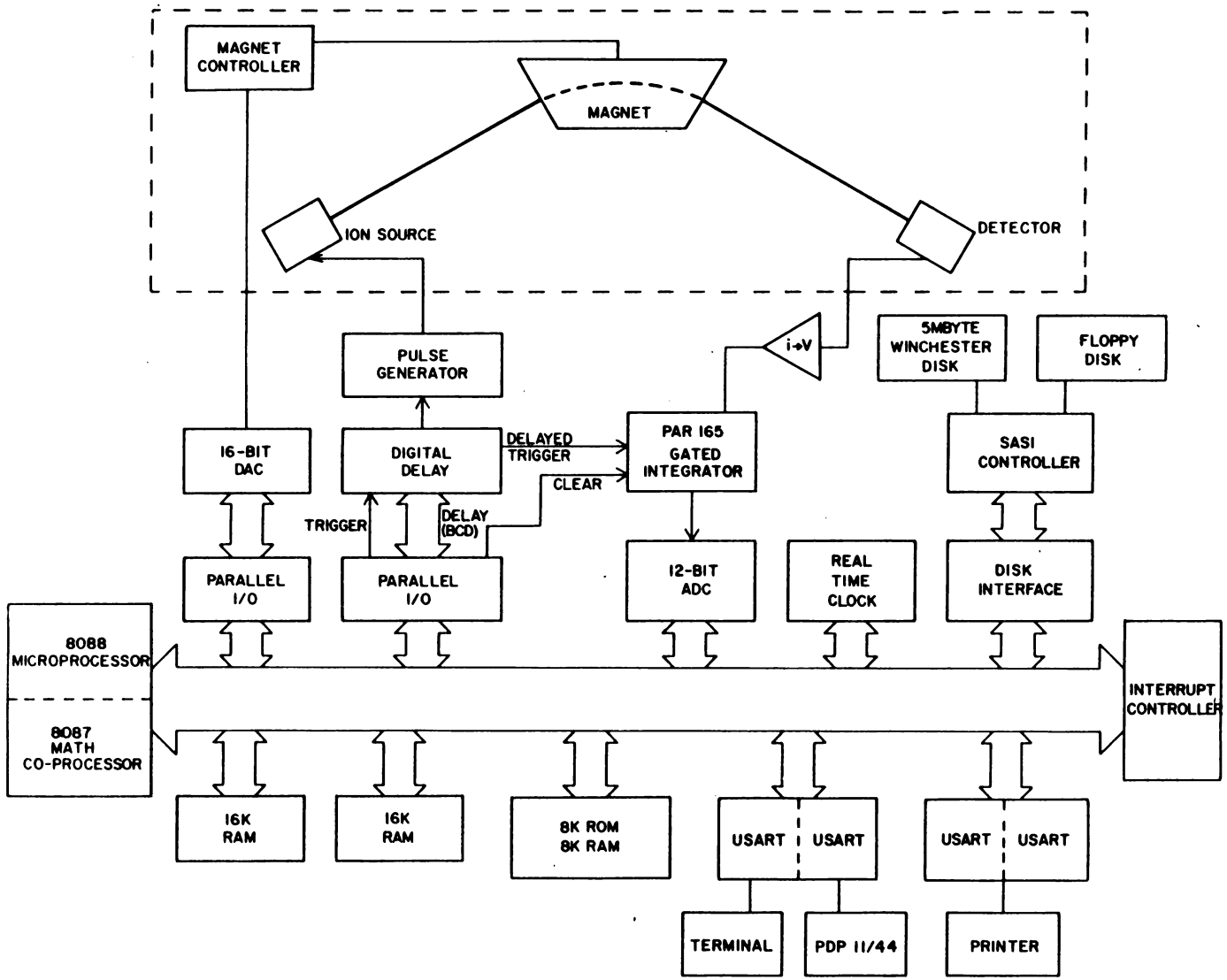

Figure 3-3 Block Diagram of the BTOF Computer System.

of the time delay the ion current at different arrival times can be measured.

Initial experiments, which measured the flight times of ions through a magnetic sector mass spectrometer, were performed with an oscilloscope to display the time-dependent ion current. These experiments confirmed the design concepts of the instrument, but to acquire the MS/MS spectra of a single compound was a prolonged task. The microprocessor system has made it substantially easier to obtain the MS/MS spectrum of a compound. The computer system is also capable of converting the time and magnetic field strength into the two mass axes for more meaningful display of the data. The increased speed with which a spectrum can be obtained will greatly assist in the modification of the instrument to improve its usefulness for multi-dimensional mass spectroscopy. In the near future, a color graphics display will be added to the system to assist in data display. This graphics display will be built on two dual-bus boards from a graphics controller module and three graphics memory plane modules. The memory modules will use 16K DRAM memory chips which will result in an 8-color 512 by 512 pixel graphics display.

SUMMARY

As these examples illustrate, the use of microcomputers for instrument control and data acquisition can greatly increase the capabilities of the analytical chemist to acquire useful chemical information. The elimination of the routine drudgery involved in data collection and data analysis permits the chemist to concentrate on the important aspects of data interpretation and experimental design. It can also increase the accuracy and precision of the experimental results and permit new, previously infeasible experiments to be performed.

CHAPTER 4 - DISTRIBUTED PROCESSING SYSTEMS -

As more experience was gained with microcomputer systems for chemical instrumentation, the systems have evolved from their early role of the automation of data acquisition into full instrument control systems. These systems have taken over the responsibility for such functions as scan generation and temperature programing from dedicated hardware controllers. The control of these functions through software instead of hardware has led to greatly increased flexibility in the operation of the instrument. An example of the increased capabilities that occur through the use of software is the generation of a mass control signal for a quadrupole mass spectrometer. The software can cause the mass spectrometer to scan rapidly until a peak is detected and then back up and scan over the peak slowly. This allows the peaks to be determined accurately while greatly reducing the scan time and thus the amount of sample needed. Another advantage of computer control of all the instrument parameters is that the computer is then able to create a data base that includes values of all of the instrument parameters along with the experimental data. This information is very helpful in

analyzing the data and allows an experiment to be rerun with the instrument in the same state as in the original experiment. As microcomputer systems were interfaced to larger more complex instruments with all of the instrument parameters under computer control, the demands on the computer system for high speed data acquisition and intelligent instrument control became greater than the capabilities of a single microprocessor. To realize fully the potential of these instruments and to permit further advances in the development of intelligent control systems, it was imperative that a more capable computer system be designed.

An obvious solution to the need for a more capable control system is the use of a larger more powerful computer such as a minicomputer. The use of a single faster computer to increase the capabilities of the control system has several drawbacks. The first problem with this approach is that the addition of more processing power in this fashion becomes more and more expensive for each increment in performance. A second problem is that the interfaces between the computer and the instrument become more complex and expensive as larger computer systems are employed. These increases in cost and complexity reduce the number of instruments that can justify the use of such control systems. An alternative solution to the need for more

computer power is the use of several small computers connected together. This results in a distributed processing environment where the different tasks needed to control the instrument are performed on several different processors. Distributed processing systems have a number of advantages for implementing advanced instrument control systems. These advantages, which are listed in Table 4-1, can be classified into three general areas: faster execution of tasks, independent execution of tasks, and increased modularity of hardware and software(34).

MULTIPLE PROCESSOR SYSTEMS

Computer systems that contain more than one processor can be classified into two general types, multiprocessor systems and distributed processing systems(35). Multiprocessor systems generally consist of several "equal" processors which share the workload of the system. During operation, a task may be assigned to any unoccupied processor. This is called dynamic load sharing because the assignment of tasks takes place during program execution. On the other hand, distributed processing systems use static load sharing, where the assignment of tasks to the

Table 4-1 Advantages of Distributed Processing Systems

Faster Execution

- Parallel execution
- Less time spent in "overhead"
- Simpler addition of hardware controllers and processors

Independent Task Execution

- Non-interference of tasks
- Elimination of task interleaving programs
- Elimination of priority assignment programs
- Simpler task program modification

Modularity of Hardware and Software

- Consolidation of related tasks
- Simpler extension of instrument capability
- Simpler debugging and troubleshooting

different processors in the system occurs during system design. Frequently the different processors in the distributed processing system are not identical, but have a variety of enhancements, such as special interfaces or numeric coprocessors, which are required for the performance of their assigned tasks. In systems with more than one processor, the processors can be said to be loosely or tightly coupled depending on the means of communication between the processors. Loosely-coupled processors communicate with each other through shared peripherials while tightly-coupled processors communicate through shared memory(36).

TASK COUPLING IN DISTRIBUTED SYSTEMS

The ways in which tasks in a distributed processing system are linked or coupled can also be categorized into several different forms. These forms of coupling are tightly-coupled tasks, loosely-coupled tasks, and terminal tasks. Tasks that have no interaction with other tasks in the system after they have been started are classified as terminal tasks. These tasks are generally assigned to intelligent peripherals in the system. These intelligent

peripherals are usually small microcomputers that are used to off-load peripheral handling tasks from the main processor. Once they have been instructed to perform a specific task, they need no further interaction with the main processor until they have completed the operation. An example of this class of task is a printer which incorporates a microcomputer to control its operation. Once the data has been transferred to a buffer in the printer the microcomputer controls the printing of all the data and then notifies the main processor when it has finished.

Tasks that involve interaction between the processors that are executing the different subtasks can be classified as loosely or tightly coupled tasks depending on the degree and time frame of the interaction. Loosely-coupled tasks involve semi-autonomous subtasks which generally interact on a millisecond time scale during the execution of the main task. An example of this type of coupling is a two processor system where one processor would acquire a number of data points from the instrument and then transfer the block of acquired data to a second processor. The second processor would then format and store the data on a disk drive while the first processor was acquiring a new block of data.

In tightly-coupled tasks, on the other hand, the

different subtasks are a much smaller portion of the overall task. This requires much more coordination between the various processors on a faster time scale. An example of tightly-coupled tasks is a system where two processors are involved in data acquisition. In this system one processor would set the instrument parameters to new values then signal the second processor that the parameters had been changed. The second processor would then acquire a data point from the instrument, signal the first processor that a point was acquired, and perform any necessary formatting operations. While the second processor was acquiring a new value, the first processor would calculate a new set of instrument parameters and, upon receipt of the signal from the second processor, would change the instrument parameters to the new values. As can be seen from this example the interaction between tightly-coupled tasks frequently occurs on the microsecond time scale. The operation of a distributed processing system is not limited to a single type of task coupling, but frequently uses all of the different types of coupling.

TASK PARTITIONING

One of the decisions that must be made during the design phase of a distributed processing system is how the different tasks in the system are going to be partitioned among the different processors. The partitioning of tasks in distributed processing systems can be performed in a variety of ways depending of the end use of the system and the critria that are used to separate tasks. Three of the major forms of partitioning are vertical partitioning, horizontal partitioning, and data access partitioning(37). Tasks that can be executed in any order or that can be executed concurently can be partitioned using horizontal partitioning. Tasks that have a definite predecessor-successor relationship can be partitioned using vertical partitioning. Data access partitioning separates tasks by what data they operate on without regard to temporal order, control, or direction of data flow. In real-time instrument control systems, vertical partitioning is usually the best choice because of the many predecessor-successor relationships involved.

The different partitions that arise from the application of vertical partitioning to real time control

systems can be classified into three major types according to the data transform concepts proposed by Yourdon and Constantine(38). The terminology for the three different types of partitions (afferent, central, and efferent) is derived from the different functions of nerve cells in the nervous system. In afferent partitions, a data input stream is converted into a form which is suitable for use by the other partitions. Operations that fall into this class are: data acquisition, signal processing, and data conversion and formatting. The tasks involved in preparing an internally generated data stream for output transmission are performed in efferent partition. These tasks involve the formatting and scaling necessary to prepare the data stream for the output interface. The central transform partitions perform all of the operations that come between the afferent and efferent partitions. This is where the main data processing operations, such as data reduction and interpretation, take place. These different classes of partitions may be utilized as criteria for assigning tasks to separate processors.

INTERPROCESSOR TOPOLOGY

Another of the decisions that must be made during the design phase of a distributed processing system is what interconnection topology should be used to link the different processors(39). Several of the more common configurations for distributed processing system are shown in Figure 4-1. Each topology has different strengths and weaknesses in such areas as, cost of implementation, complexity of the interprocessor interface, communication bandwidth, and the ease with which another processor can be added. The loop configuration has a low to moderate level of complexity and can be implemented at a low to moderate cost. In this system, a new processor can be added quite easily by connecting just two communication paths. The communication bandwidth is low to moderate with the loop configuration because a message from a sending processor to a receiver must pass through all of the intervening processors. An example of a low cost version of this topology is the HP-interface loop (HP-IL)(40), which was designed to interconnect programmable calculators and their peripherals.

The star configuration, which consists of a central

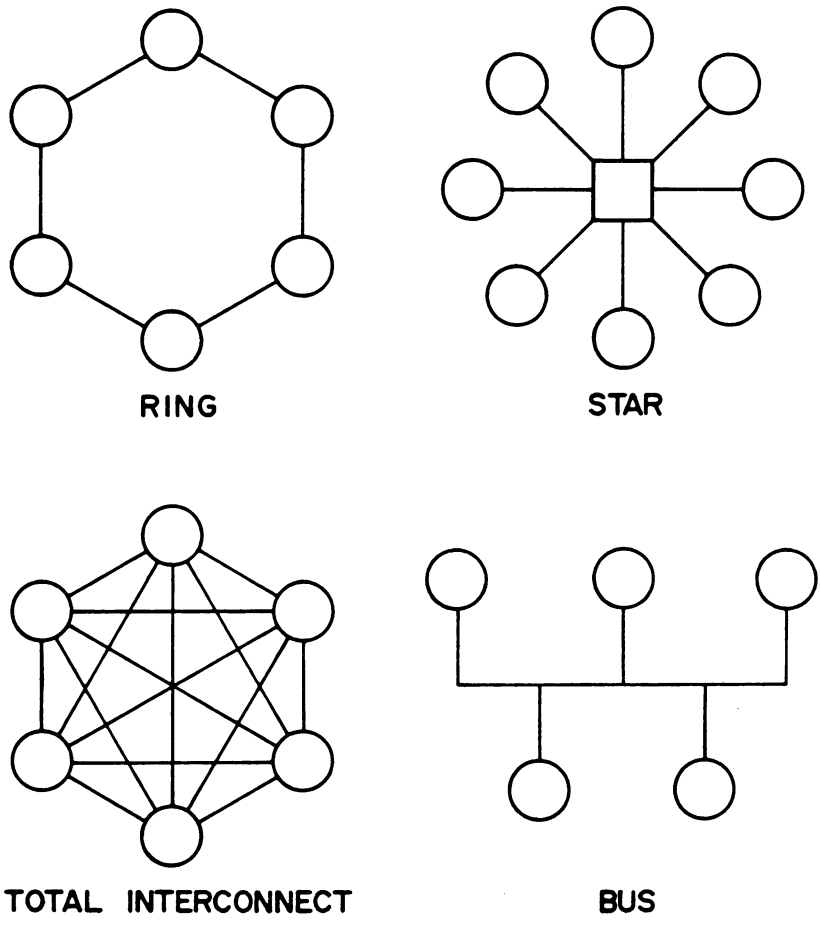

Figure 4-1 Distributed System Topologies.

switch that is connected directly to each of the processors in the system, has a moderate cost of implementation and a moderate to high complexity. This system has a moderate to high communication bandwidth limited by the bandwidth of the central switch. The IBM network/440, which connects several remote 360 computers together through a central 360 mainframe computer, is an example of this topology. This topology is also frequently used to connect a central mini or main frame computer to a number of satellite processors, where the communication paths are used to upload data from the satellite processors to the host instead of being used for communications between different satellite processors.

The topology which has the highest cost and complexity is the total interconnect configuration where each computer has a direct link to every other computer in the system. The addition of another processor to the system is difficult since the new processor must be connected to each of the existing processors. The advantage of this topology is that it has a very high communication bandwidth due to the dedicated paths between the processors. An example of the type of system is a fully interconnected version of the IBM attached support processor system(41).

The fourth topology that is shown in Figure 4-1 is the common bus system which has a low to moderate cost and

complexity. In this system it is easy to add another processor and the resulting system has a moderate communication bandwidth which is limited by the bandwidth of the bus. The IEEE-488 GPIB instrumentation(42) and the military 1553 avionics bus(43) are examples of the common bus topology.

INTERPROCESSOR COMMUNICATION MODES

To facilitate the development of advanced control systems for complex instruments, the decision was made to develop the modules necessary to interconnect single processor systems, built from the standard modules, together to form a distributed processing system. To implement this system, it was necessary to define the different modes of communication between the processors in the system. An examination of the tasks involved in real time control systems revealed the need for four modes of interprocessor communication. These modes are block data transfer, task assignment, parameter transfer, and task coordination. The block transfer of data is utilized to download programs into the different processors and to move data sets between processors. The task assignment mode is

used to inform the different processors in the system what tasks they should be performing and in what order. The parameter transfer mode is used to transfer small amounts of information between the processors. These parameters are frequently used in conjunction with task assignment where they supply such information as the number of times a command should be executed or the limits that the command should utilize. The final mode of interprocessor communication involves the transfer of status information between the different processors so that tasks on the different processors can be coordinated with each other.

DISTRIBUTED PROCESSOR DESIGN GOALS

To help insure that the distributed processing system would meet the requirements of real time instrumentation, the list of design goals given in Table 4-2 was compiled. The first of these design goals is that the different modes of interprocessor communication be supported by dedicated hardware paths between the different processors. Although the interprocessor communication modes could have been implemented with just some shared memory, the use of dedicated hardware paths increases the speed and

Table 4-2. Distributed Microprocessor System Design Goals

1. Hardware Support for the Interprocessor Communication Modes

   Block Data Transfer

   Task Assignment

   Parameter Transfer

   Task Coordination

2. Non-interference by Interprocessor Communications with Timely Events

3. No Resident Monitor Required in Auxiliary Processors

4. Modular Hardware for easy Adaptability

5. Easy Transition from a Single Processor System to a Distributed Processing System

capabilities of the interprocessor communications. One of the capabilities that the dedicated hardware paths provide is non-interference with timely events which is the second design goal. This is an important feature because tasks frequently occur in real time systems that are time-critical and must not be disturbed. To meet this goal, the paths between the processors were designed so that commands and parameters could be transferred to a processor and status information exchanged between processors without interrupting task execution. Although the block data transfer could have been designed so that it was non-interfering, this was not necessary because these transfers normally take place either at system startup time when programs are loaded into the processor or when the processor requests that a block of data be moved from it's memory. In either situation the processor is not executing a time critical task. Allowing this mode to interfere with the processor greatly simplifies the design of the hardware needed to support this mode of communication.

The topology that was chosen for this system was the common bus topology. The decision to use this topology was made because of the ease with which a new processor may be added and the moderate communication bandwidths that this topology provides. The system was designed to connect together up to eight processors constructed from the

standard modules that had been designed for single microprocessor systems. In this distributed processing system the "master" processor controls the interprocessor communications paths between itself and the rest of the "slave" processors.

Three different hardware paths were implemented to support the four modes of interprocessor communication listed in Table 4-2. These three paths are direct memory transfer, command transfer, and status transfer. The direct memory transfer path supports the block data transfer mode of interprocessor communication. It allows the bus master, a processor called the communicator, to read from or write to memory in any of the slave processors. This path does not require that the slave processor be operating to transfer data. Thus, processors that are completely RAM based can be down-loaded with code at system startup. This results in increased flexibility of the software since any program can be easily changed and then reloaded into the slave processor. This path also allows new interfaces on a slave to be tested from the communicator to determine if they are operating correctly. This feature is especially helpful when a pathological error exists that prevents the operation of the slave processor. The second hardware path is the command transfer path which is used to perform both the task assignment and parameter passing modes of

interprocessor communication. This path is implemented by a set of first-in first-out (FIFO) buffers on each of the slave processors into which the bus master, called the coordinator, can load both parameters and commands for the slave processors. The command transfer path also has some immediate functions which lets the coordinator reset, hold, or interrupt any slave processor. The third path implemented is the status transfer path which supports the task coordination mode of communication. A bank of dual-ported memory on each processor is used to implement this path. Over a separate part of the interprocessor bus, information on the hardware and software status of each processor is written into the dual-ported memory on every processor. This allows a processor to determine the status of any processor by merely reading the corresponding status information from it's dual-ported memory.

The interprocessor hardware was designed so that the communicator and the coordinator could be either two separate processors or combined on a single processor. The decision whether to use a separate coordinator and communicator or a combined master processor depends on the nature of the instrument that is to be controlled. If the control system must respond to a number of asynchronous events, separate coordinator and communicator processors are preferred, since the separate processors allow more

processing time to be spent on monitoring the status of events and issuing appropriate commands. However, if the instrument is controlled in a synchronous manner (e.g. the control system sends a stimulus to the instrument and then measures the result of the stimulus), a combined coordinator/communicator (master) processor is probably the preferred implementation due to the simpler software that is needed to control the distributed processing system.

CHAPTER 5 - INTERPROCESSOR HARDWARE -

The three interprocessor communication paths were implemented as a set of dual height modules that connect to both the local bus and the top bus which is then used as an interprocessor bus. The arrangement of these modules is illustrated in Figure 5-1 which shows a typical three processor system. The function of each processor in the distributed processing system (communicator, coordinator, or slave) is determined by the types of modules that are mounted on the interprocessor dual bus board. These different modules and the interprocessor bus are discussed in the following sections.

THE INTERPROCESSOR BUS

The top bus traces on the interprocessor dual bus boards are jumpered out to the backplane where a 50 pin ribbon cable is used to connect the interprocessor bus to all of the processors in the distributed processing system. The pinout of the interprocessor bus cable is listed in

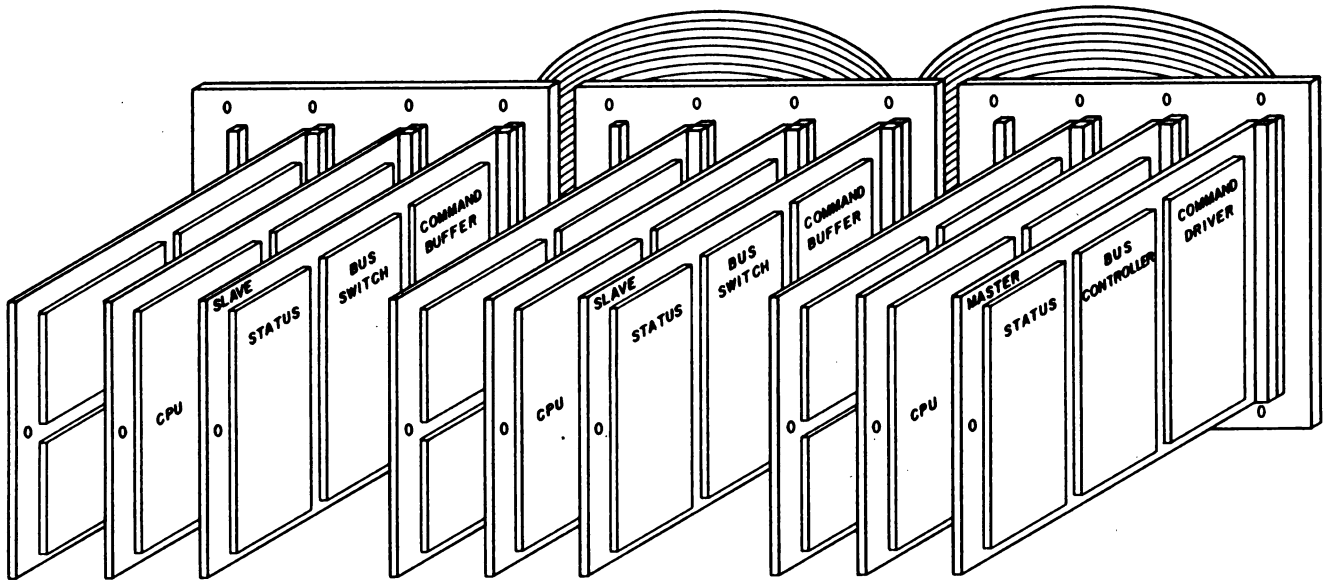

Figure 5-1 A Typical Distributed Processing System.

Table 5-1, and the definition of the signals are given in the following list.

ID0-ID7 Eight bidirectional interprocessor data lines that carry the data during a direct memory transfer and carry 8 bits of the command during a command transfer.

IA0-IA15 Sixteen interprocessor address lines that carry the address information during a direct memory transfer and the remaining 16 bits of command during a command transfer.

IA16-IA18 The three high-order interprocessor address lines that carry the number of the processor being accessed during direct memory and command transfers.

IRD\ An active LO signal that is used by the communicator to enable the transfer of data during an interprocessor read operation.

IWR\ An active LO signal that is used by the communicator to enable the transfer of data during an interprocessor write operation.

IIO/M The communicator drives this signal LO during an interprocessor memory transfer.

IS1\ This signal is driven by the communicator to indicate the direction of data transfer during a direct memory

Table 5-1. Pinout of the Interprocessor Bus Cable

PIN 1	IP GND	PIN 2	STGND
3	STB\	4	STBEN\
5	SA3	6	SA2
7	SA1	8	SA0
9	SD7	10	SD6
11	SD5	12	SD4
13	SD3	14	SD2
15	SD1	16	SD0
17	IRDY	18	ICMD
19	IA18	20	IA17
21	IA16	22	ILOCK
23	IWR\	24	IRD\
25	IIO/M	26	IS1\
27	IA15	28	IA14
29	IA13	30	IA12
31	IA11	32	IA10
33	IA9	34	IA8
35	IA7	36	IA6
37	IA5	38	IA4
39	IA3	40	IA2
41	IA1	42	IA0
43	ID7	44	ID6
45	ID5	46	ID4
47	ID3	48	ID2
49	ID1	50	ID0

transfer. A LO level indicates that the data is to be transferred to the communicator.

ILOCK A HI level on this line, which is driven by the communicator, indicates that the addressed processor is not to be released at the end of the direct memory transfer operation.

IRDY A LO level on this line keeps the communicator in a wait state during a direct memory transfer until the addressed processor releases its local bus.

ICMD The coordinator drives this signal HI to strobe the command information into the FIFO buffers on the addressed processor during a command transfer.

SD0-SD7 Eight status data lines that carry the status information between all of the processors.

SA0-SA3 Four status address lines that are used to control which byte of status information is being written into the dual-ported memory on all of the processors. These four signals are driven by the coordinator.

STB\ A LO level is driven on this line to strobe the data on SD0-SD7 into the dual-ported memory on all of the processors.

STBEN\ This signal is used to establish the timing of the

STB\ signal. It is driven LO by the coordinator to enable the STB\ signal.

STGND This is a reference ground for the status bus signals.

IPGND This is a reference ground for the rest of the interprocessor bus signals.

In addition to the signals on the interprocessor bus, the dual bus board provides several uncommitted traces that are used to connect signals between interprocessor modules on the same processor. These signals provide the processor number (PN0-PN2), five chip selects (CB1\-CB4\,BCREG), and three hardware status signals (BUFE\, BUFF\, CMDHLD).

DIRECT MEMORY TRANSFER PATH

The module in each processor that carries out the direct memory transfer path is the bus controller or the bus switch module depending on whether the processor is the communicator or one of the other processors. The block diagrams of each of these two modules are shown in Figure 5-2. The bus controller module was originally designed to be used with an 8085 CPU. However, it was later modified

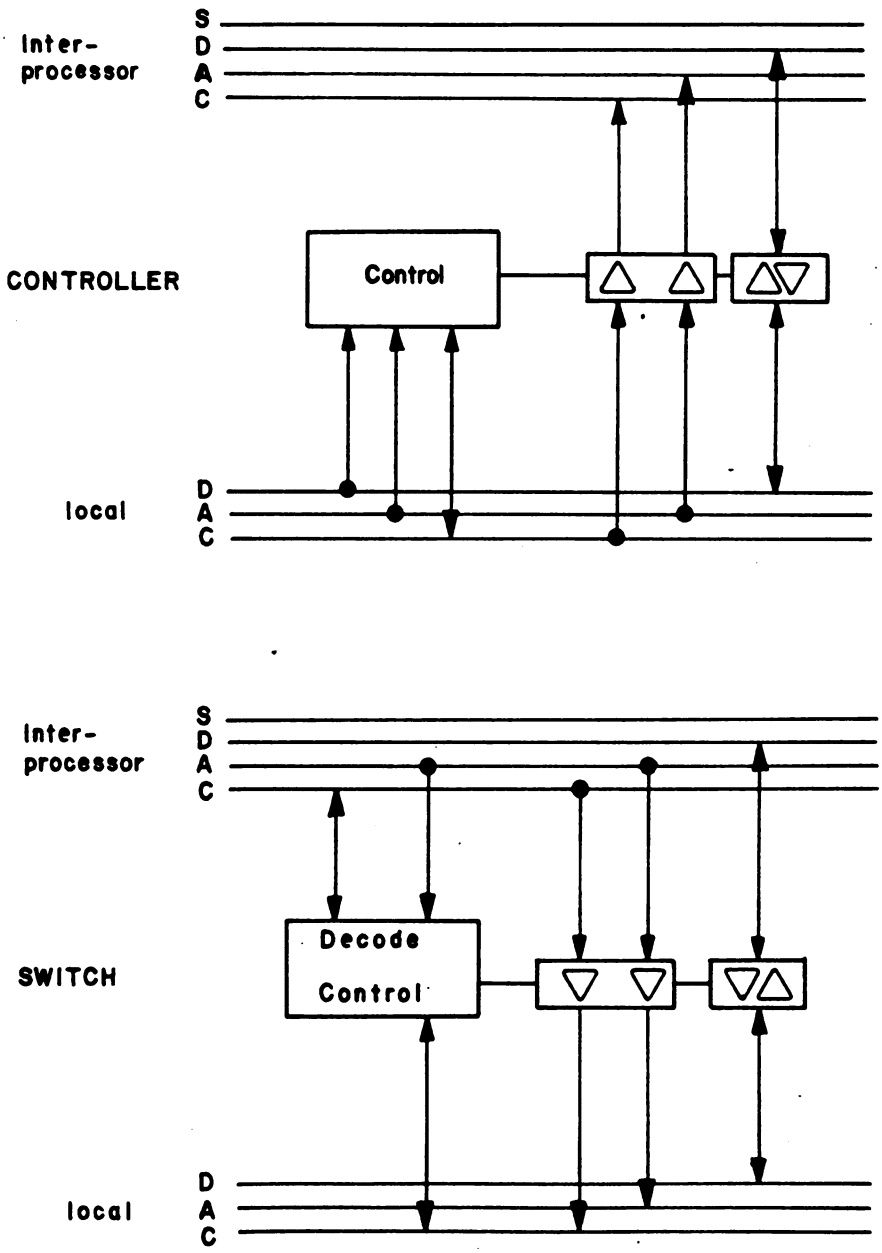

Figure 5-2 Block Diagram of the Bus Controller and Switch.

to operate with the 8088 CPU because the 8088 CPU's increased address space permits more efficient operation of the direct memory transfer path. In the communicator, the top half of its 1 megabyte address space is defined to be interprocessor memory space. Each of the slave processors, which are restricted to 64 Kbytes of memory space, is assigned a unique eighth of the available 512 Kbytes. Any read or write operation to these addresses in the communicator initiates an interprocessor memory transfer and results in an actual read or write in the corresponding byte in the slave's memory. Thus the top half of the communicator's memory is actually the combined memory of the slaves. The main stages in this transfer are: the bus controller drives the interprocessor bus, the addressed bus switch holds the slave processor and takes control of the slave's local bus, the communicator performs a read or write operation to the slaves memory, and the slave processor is released from the hold state.

There are two different options which can modify the stages in the basic transfer. The first is that the communicator can set a latch which then generates the **ILOCK** signal during the next interprocessor transfer. This signal causes the bus switch on the addressed slave system to keep the slave processor in a hold state after the transfer is finished. Subsequent transfers to this slave

can then take place at a faster rate because there is no need to wait for the processor to finish the current bus cycle and release the local bus. The lock latch is cleared before the last transfer to the slave system, which allows the slave processor to be released from its hold state. This option is useful for batch transfers such as loading blocks of code or moving blocks of data from the slave. The second type of deviation from the basic transfer takes place when the interprocessor transfer takes longer then a preset time. If this happens, a bus "timeout error" occurs which immediately terminates the interprocessor transfer and generates an interrupt to the communicator processor. This error can take place if a peripheral controller (DMA controller) in the slave system has control of the local bus and won't release the local bus for the bus switch to use.

The schematic diagrams of the bus controller and the bus switch are given in Figures 5-3A and 5-3B and Figures 5-4A and 5-4B respectively. These diagrams will be used to discuss the steps that take place during a typical interprocessor transfer between the communicator and a slaves memory:

- If the communicator desires to perform a block transfer it sets the LOCK flip/flop.

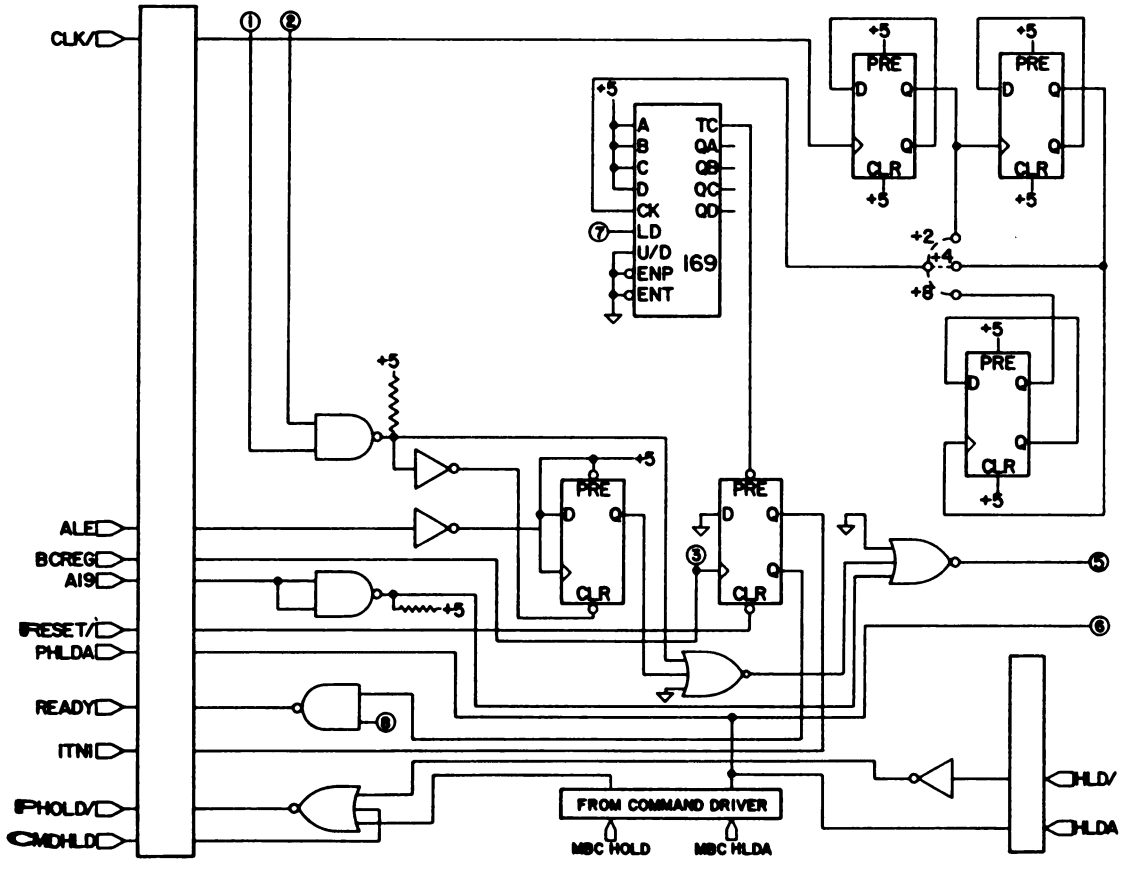

Figure 5-3A Schematic of the Bus Controller.

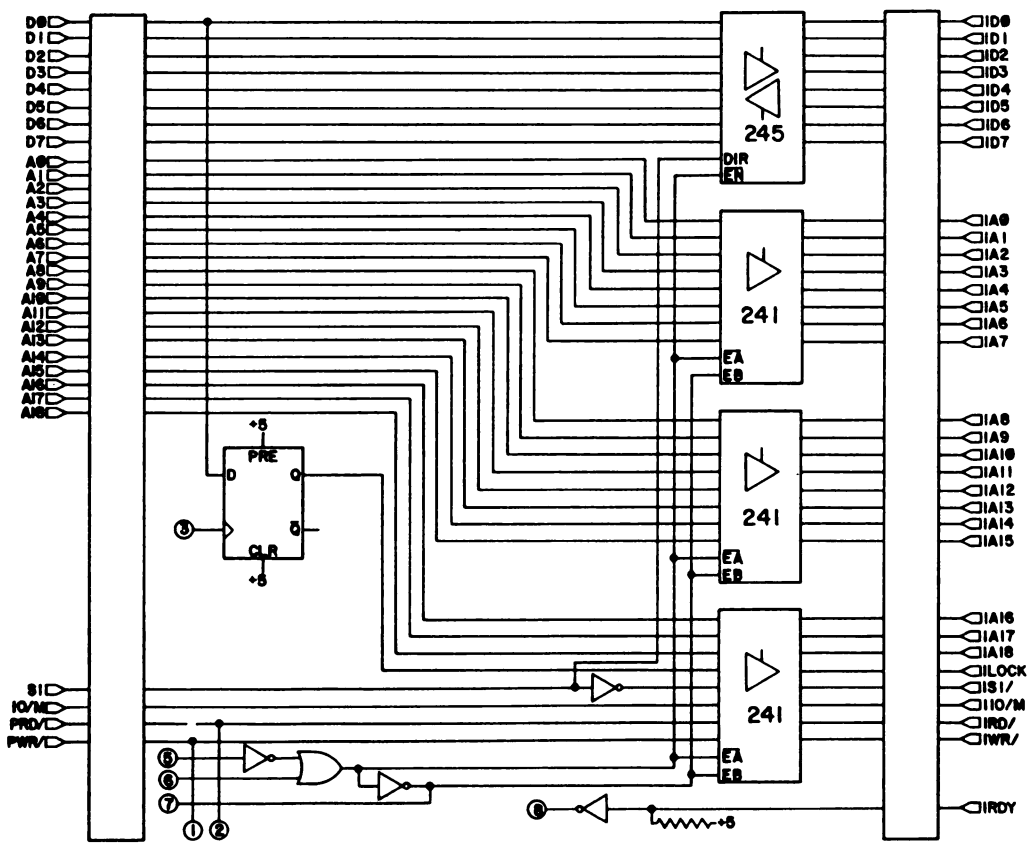

Figure 5-38 Schematic of the Bus Controller.

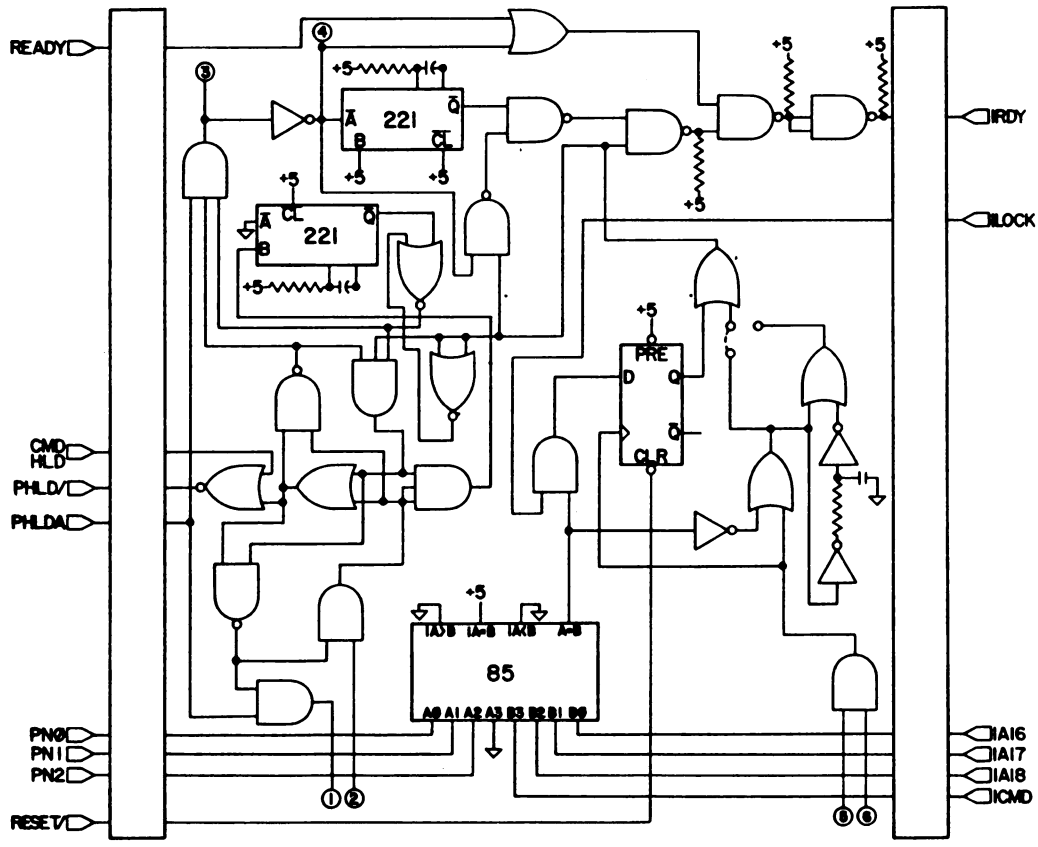

Figure 5-4A Schematic of the Bus Switch.

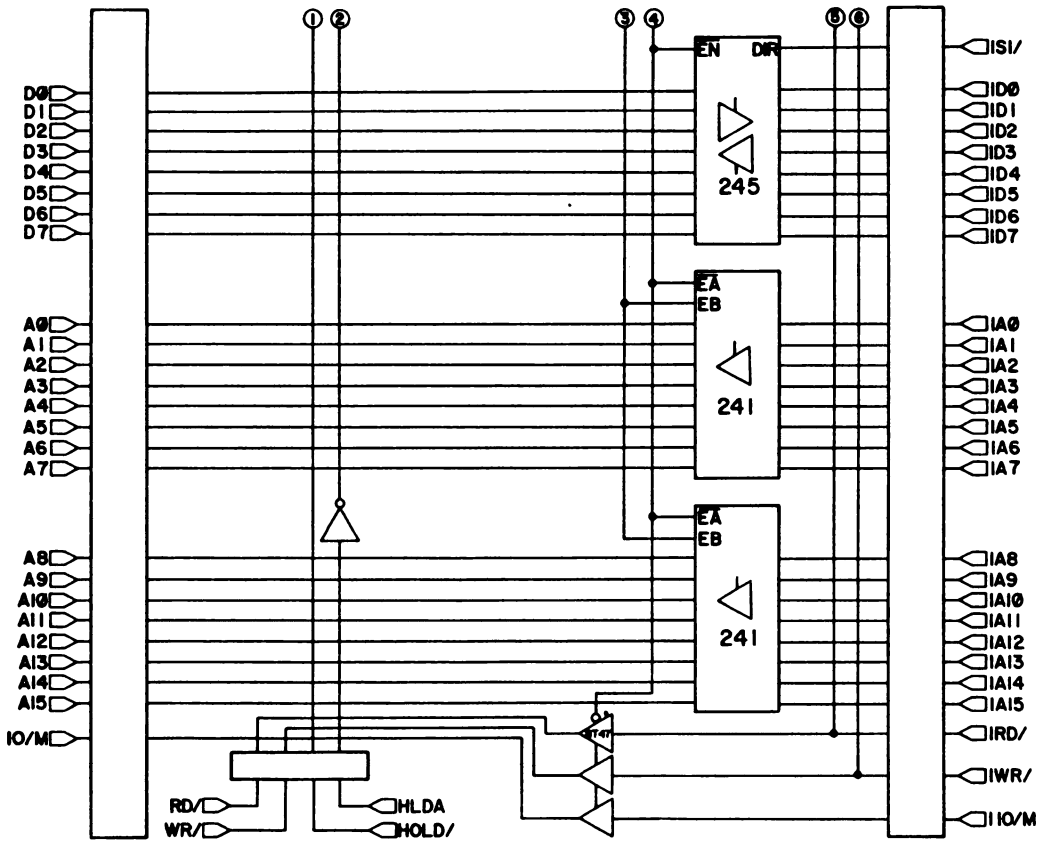

Figure 5-4b Schematic of the Bus Switch.

- The communicator addresses a memory location which corresponds to the desired location in the slave processor memory space.
- Logic on the 8088 cpu module inserts one wait state which gives the interprocessor modules time to respond.
- The control logic in the bus controller recognizes that a slave address is being addressed by combining A19, ALE, PRD\, PWR\, and PHLDA.
- The control logic then enables the tri-state drivers which drive the data, address, and control signals onto the interprocessor bus.
- When the bus switch in the slave processor recognizes that it is being addressed by combining IA16, IA17, IA18, ICMD, IRD\, and IWR\ it drives IRDY LO, which causes RDY on the communicator to be driven LO thus placing the processor into a wait state. It also generates a signal which is combined with HOLD\ in an arbitration circuit to drive PHOLD\ LO. This requests the use of the local bus from the slave processor.
- After the slave processor has finished any current bus cycle it releases the local bus.
- The slave processor then informs the bus switch that it may assume control of the bus by driving PHLDA HI.
- The bus switch then enables a set of tri-state drivers which place the address, data, and control information

onto the local bus.

- After sufficient setup time has passed the communicator processor is released from the wait state by driving IRDY HI.
- The communicator then completes the read or write operation and disables all of the tri-state drivers.
- If ILOCK was LO during the transfer then PHOLD\ is driven HI which releases the slave processor from its hold state and permits it to resume execution. If ILOCK was HI during the transfer then PHOLD\ is kept LO and the processor stays in a hold state.

If the HOLD\ signal on the slave processor was active when the transfer starts, the arbitration circuit on the slave processor never allows the IRDY signal to return HI. This permits a counter on the bus controller to reach terminal count and set a flip/flop. When this flip/flop is set, it causes the RDY signal on the communicator to be driven HI, which releases the processor from the wait state and generates an interrupt to the processor.

COMMAND TRANSFER PATH

The command transfer path is implemented by a command driver module on the coordinator and command buffer modules on all of the other processors. A block diagram of these two modules is given in Figure 5-5. These modules use the data and address lines of the interprocessor bus to transfer a 24-bit command to the FIFO buffers on the desired slave system. The FIFO buffers, which are located on the command buffer modules, allow queuing of up to 32 commands for the slave processor. Figures 5-6A and 5-6B are schematics of the command driver and Figures 5-7A and 5-7B are schematics of the command buffer. The stages that take place during a typical transfer are:

- The coordinator loads the processor number of the slave to which the command is to be sent into the register controlled by chip select CB4\.
- The coordinator loads 16 bits of the command into two 1-byte latches controlled by chip selects CB1\ and CB2\.
- The coordinator then writes the remaining eight bits of the command to a fourth address (CB3\).
- When CB3\ goes LO the command driver module drives RDY

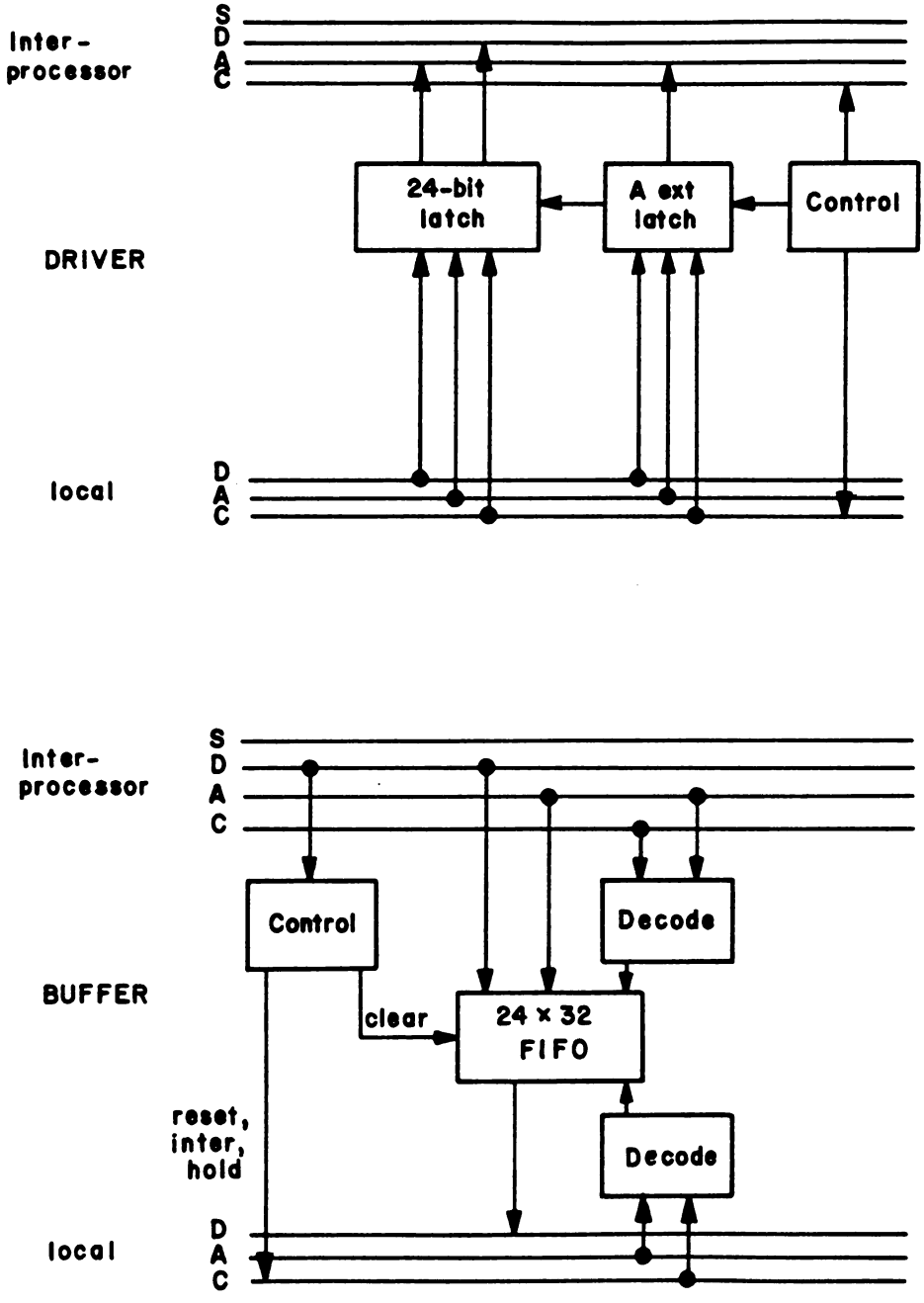

Figure 5-5 Block Diagram of the Command Driver and Buffer.

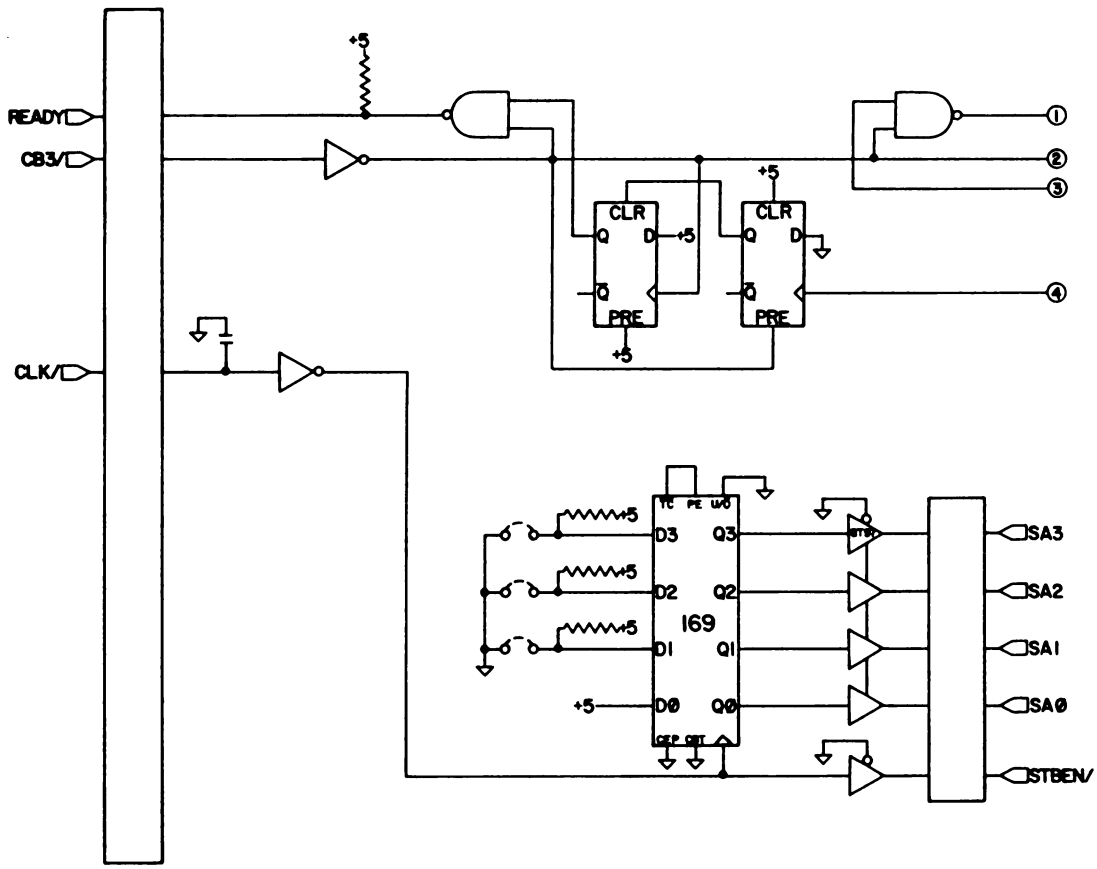

Figure 5-6A Schematic of the Command Driver.

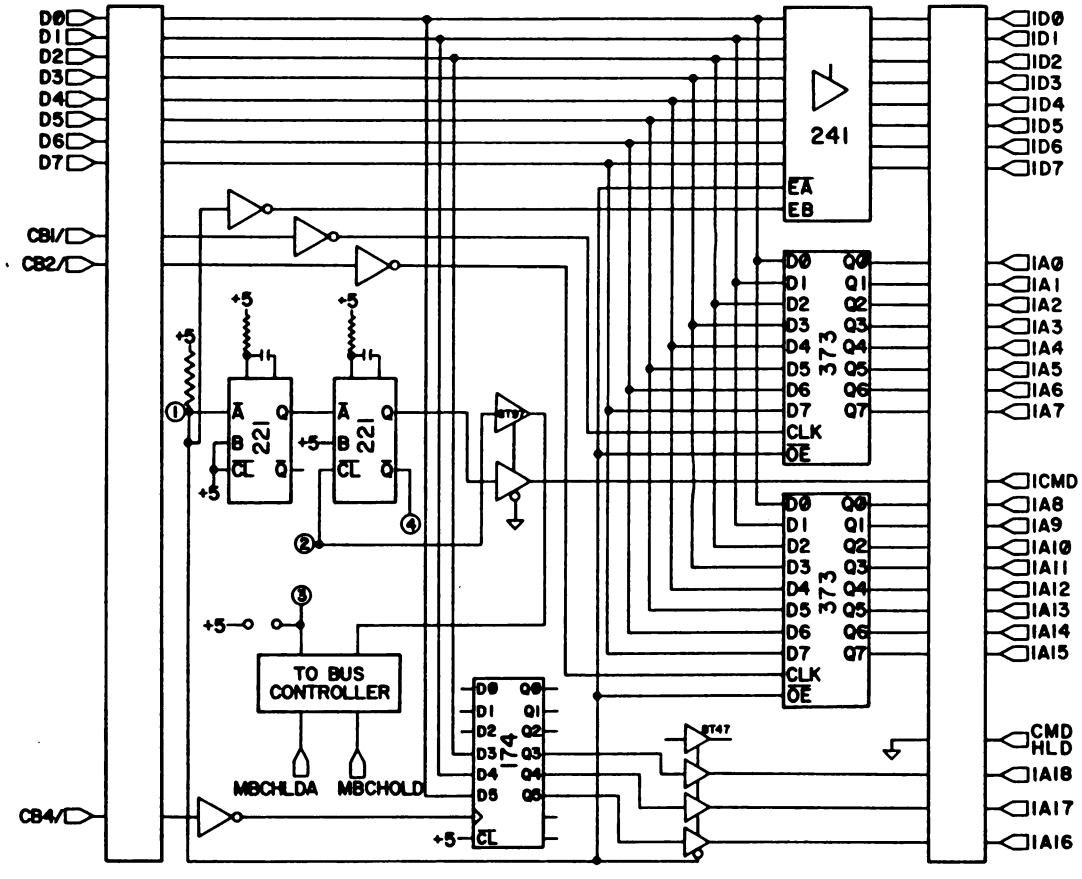

Figure 5-6B Schematic of the Command Driver.

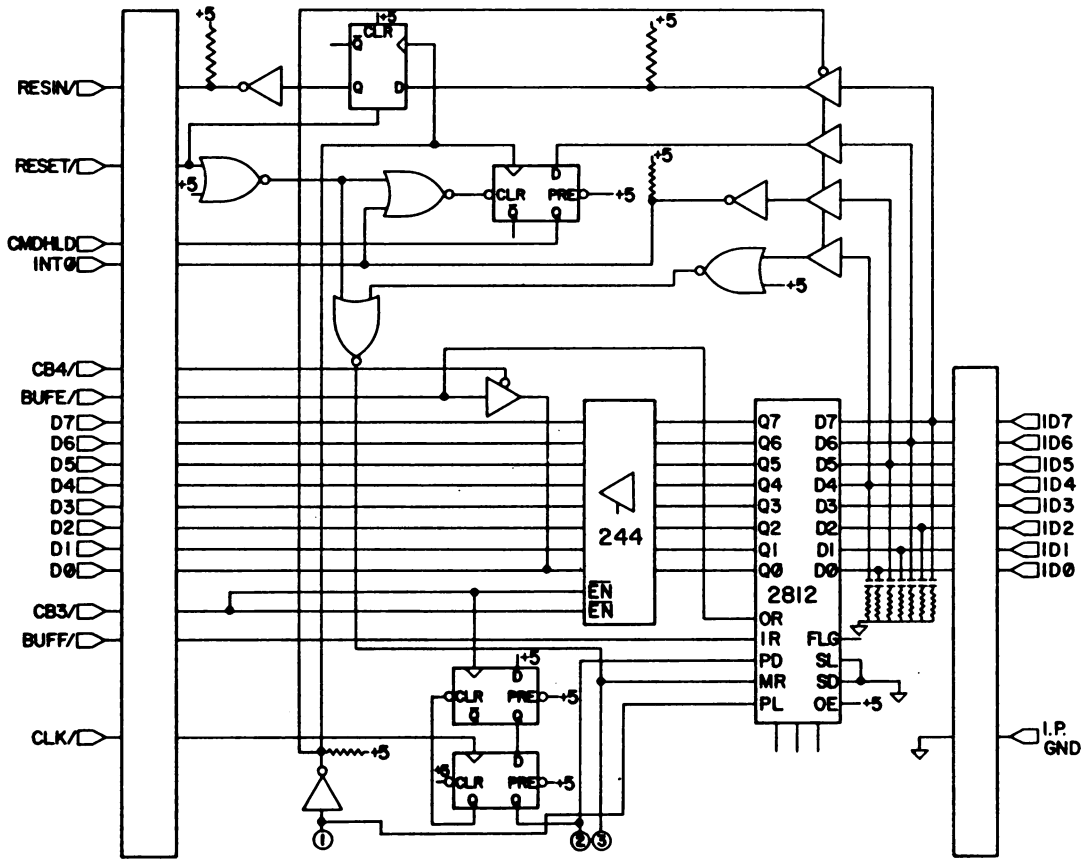

Figure 5-7A Schematic of the Command Buffer.

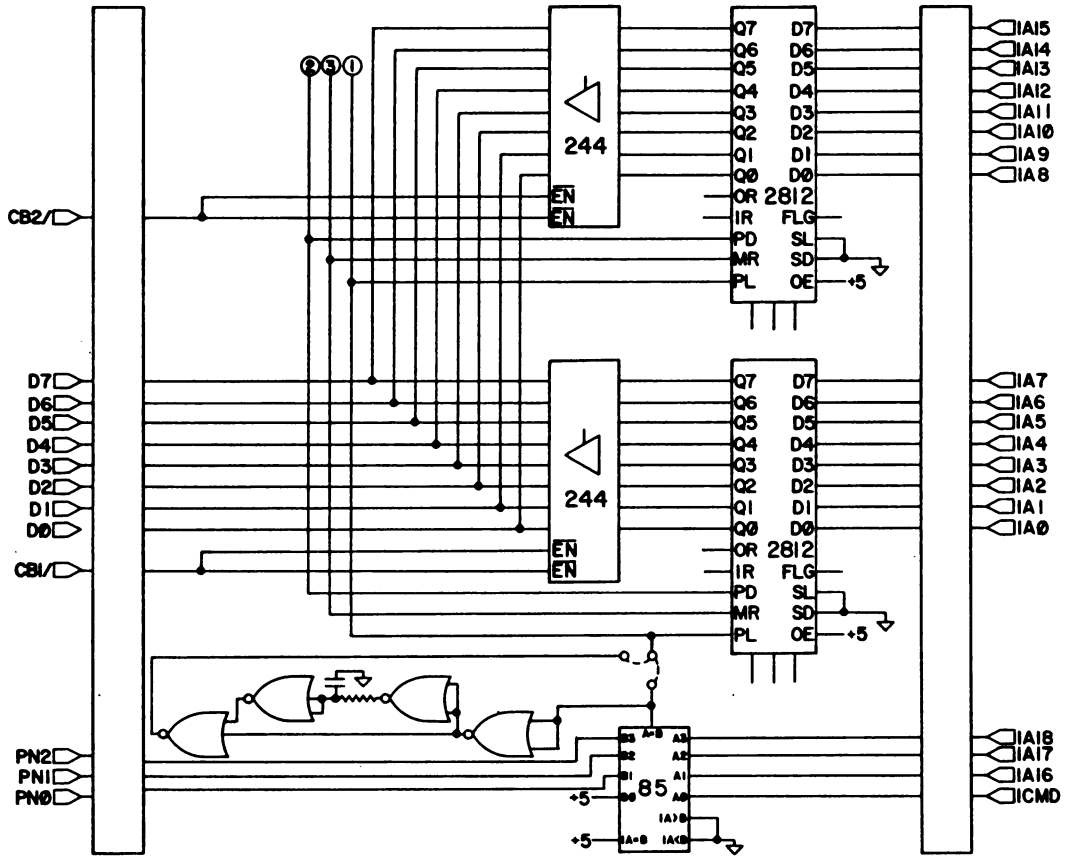

Figure 5-7B Schematic of the Command Buffer.

LO which places the coordinator in a wait state. It then requests the use of the interprocessor bus from the communicator by placing the communicator in a hold state. The coordinator then waits for a hold acknowledge from the communicator. If the system uses a combined coordinator /communicator, then the hold and the hold acknowledge signals are jumpered to appropriate levels.

- When the command driver is granted use of the interprocessor bus, it drives the 24 bits of the command and the specified slave number onto the interprocessor bus. After a setup time has passed, a HI pulse on the ICMD signal is generated.
- The control logic on the command buffer module loads the command into the FIFO buffers when it recognizes it's slave number and a HI on ICMD.
- The coordinator is then released from its wait state and control of the interprocessor bus is returned to the communicator.

The commands are loaded into the FIFO buffers without interfering with the slave processor. The slave processor can check if a command has been sent to it by reading from the address that generates CB4/. If bit Ø is HI, a command is in the FIFO buffers. The slave processor can then read the command by reading from addresses that generate CB1/,

CB2/, and CB3/. Reading from CB3/ also causes the next data in the FIFO buffers to become available. In addition to being loaded into the FIFO buffers, four of the bits in the command have immediate functions. The first of these immediate functions enables the coordinator to clear the FIFO buffers on the command buffer module. This allows the coordinator to empty the buffers at startup time or to cancel commands it has already sent to the command buffers. The second immediate function is the ability to reset the slave system. This is accomplished by driving the RESIN\ signal LO, which allows the coordinator to initialize the entire distributed processing system to a known state by resetting all of the slave processors. The third function is the ability to hold the slave processor by driving the CMDHLD signal HI. This function can be used at startup before code has been down-loaded into the slave to keep the slave from executing random instructions. The final immediate function is the ability to issue an interrupt to the slave; this could be used to cause the slave processor to read the command buffers before the current task is finished. A high priority command could thus be executed before it's normal turn.

STATUS TRANSFER PATH

The status module, which is the same for each processor, contains sixteen bytes of dual-ported memory, a one byte software status register, and an octal hardware status driver. A block diagram of the status module is shown in Figure 5-8. One of the ports of the dual ported memory is written into by a hardware driven status bus, which is part of the interprocessor bus, while the other port can be read by the processor using the local bus. A schematic diagram of the status module which implements the status transfer path is shown in Figures 5-9A and 5-9B.

Each of the processors is assigned two addresses on the status bus, one for it's software status byte and one for it's hardware status byte. A counter, which is on the command driver module, cycles the address lines on the status bus (SA0-SA3) through the addresses that are assigned to the different processors. When a status module recognizes an address that matches one of its status bytes, it drives that byte onto the status data lines SD0-SD7. The status module then waits for STBEN\ to go LO at which time it drives STB\ LO. This causes the information on lines SD0-SD7 to be written into the corresponding address in all

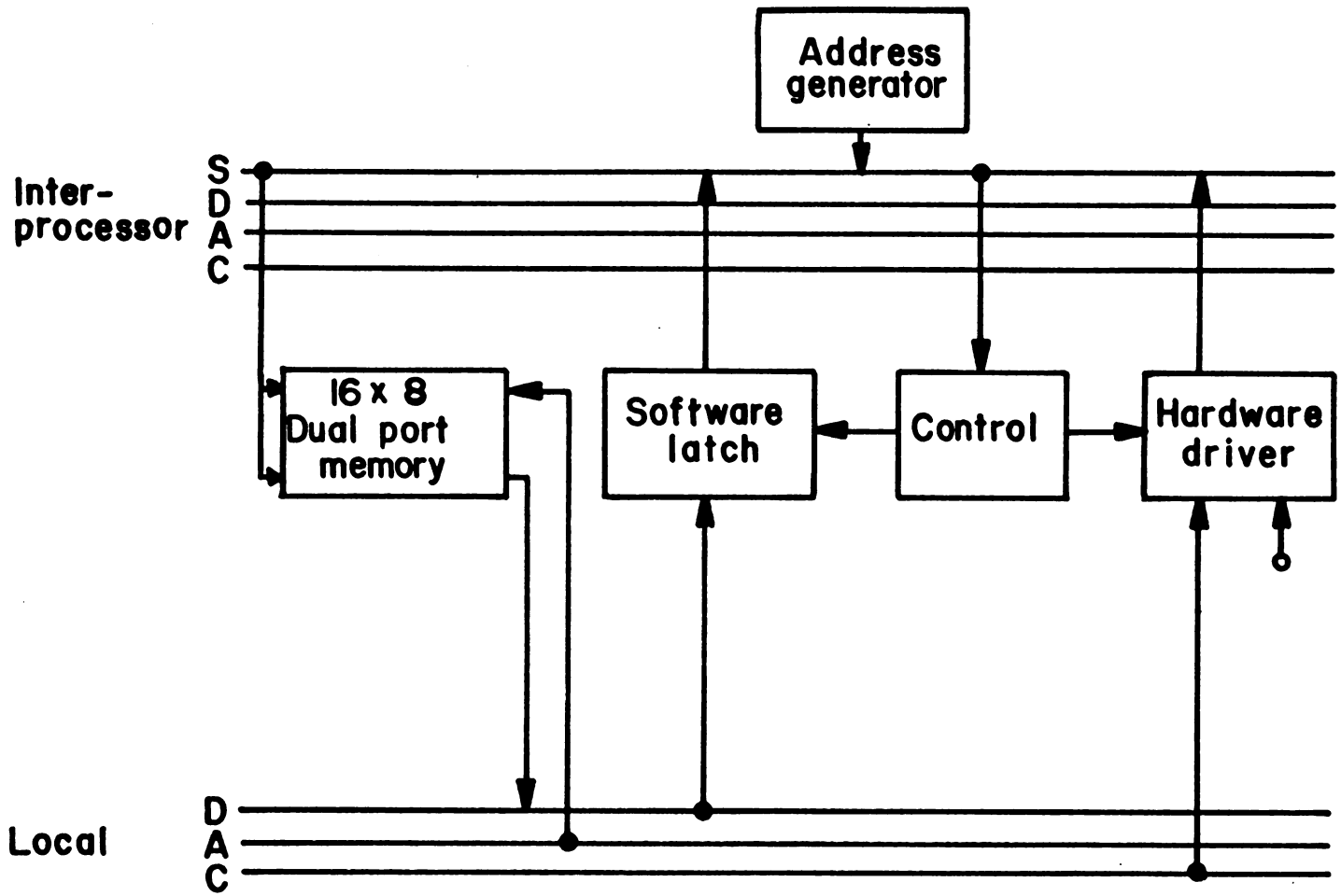

Figure 5-8 Block Diagram of the Status Module.

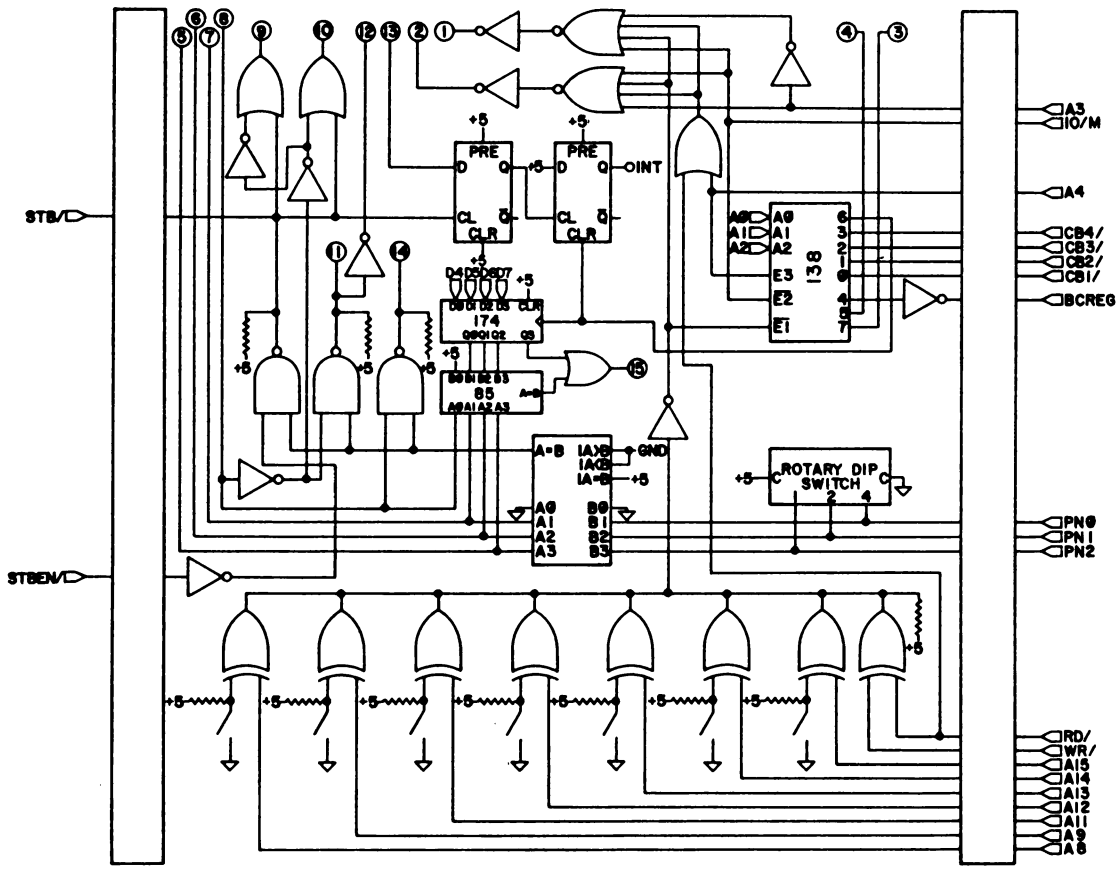

Figure 5-9A Schematic of the Status Module.

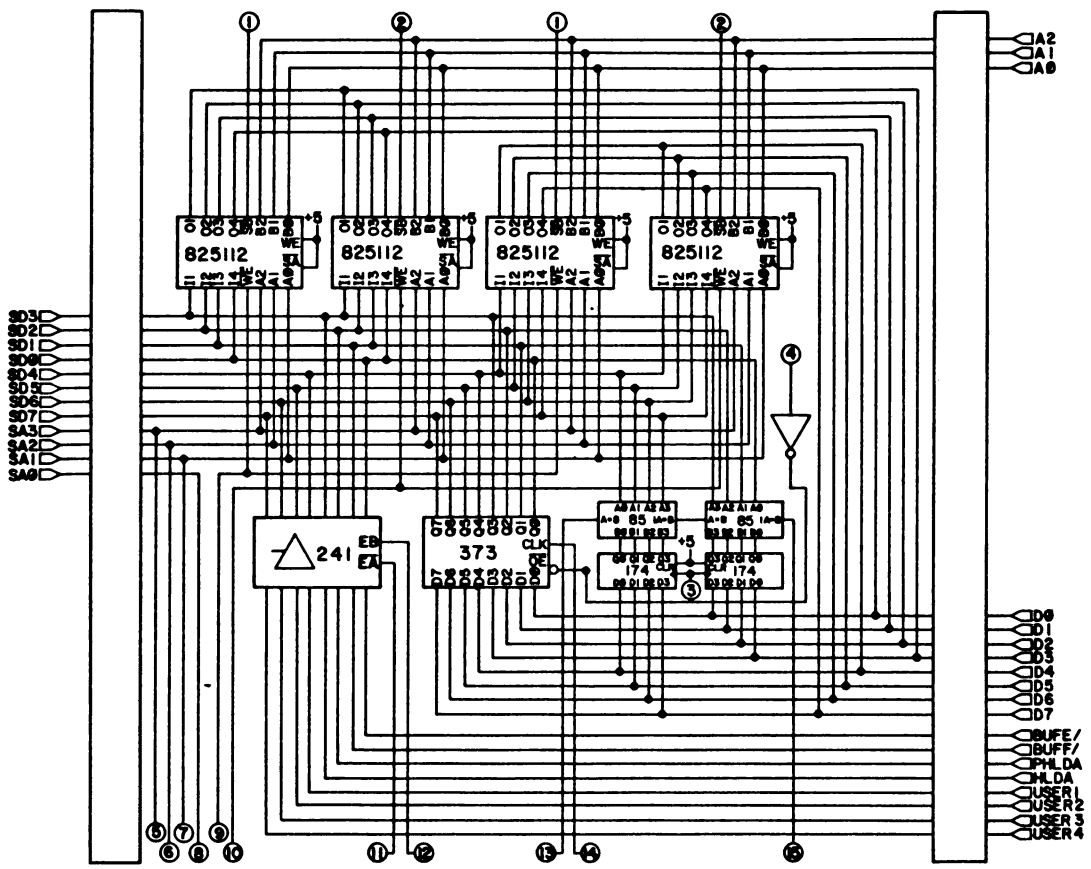

Figure 5-9B Schematic of the Status Module.

of the dual-ported memories. The hardware that generates the addresses for the status bus can be jumpered so that fewer than sixteen address are generated. This allows the status information to be update more rapidly if there are less than eight processors in the system. When a processor wants to know the status of any processor in the system, all it has to do is read the corresponding byte in it's dual ported memory

Four of the bits in the hardware status byte are available for definition by the user while the other four have been predefined. The four predefined bits are command buffers empty (BUFE\), command buffers full (BUFF\), processor on hold (PHLDA), and local bus controlled by a peripheral controller (HLDA). These signal lines are used mainly by the coordinator and communicator to control the exchange of information between processors. The software status byte is used to synchronize the execution of tasks on the different processors in the system by exchanging information about what task is being executed by each processor.

The status module is also equipped with a set of latches and comparators that monitor the status data and address lines. This circuit is capable of generating an interrupt to the local processor in two ways, first if the

software status byte of any processor matches the value stored in the latches or second, if the software status byte of a specific processor matches the value stored in the latches. This allows a processor to perform a task while waiting for a specific event on another processor. The status module also has the address decoding logic on it for all of the interprocessor modules.

PERFORMANCE

The combination of these interprocessor paths allows distributed processing systems to be designed that can effectively implement real time control systems. As an example of the type and complexity of system that can be built with these modules, a control system for an advanced mass spectrometer is examined in the next chapter. This system, which uses a 5 MHz 8088 processor as a combined coordinator/communicator, was used to determine the communication rates of the different hardware paths. The direct memory transfer path was capable of transferring 270 Kbytes per second between the master and a slave using the unlocked mode. If the locked mode of communication was employed the transfer rate increased to 345 Kbytes per

second. This system used four processors which results in a status transfer latency of 1.6 microseconds.

The transfer rate for the command transfer was determined using some high level FORTH routines which could be implemented faster in assembly language. These FORTH routines resulted in a command transfer rate of 3,400 commands per second if the FIFO full bit in the hardware status byte was checked. The communication rate increased to 7,600 commands per second if the check for FIFO full was not performed.

These communication rates are sufficient for many real time applications with the status transfer path needing the most improvement due to the amount of time that the interpretation of the status byte needs in software. This improvement might be accomplished by the implementation of a status transfer mode that would allow interrupt signals to be transferred between processors (see the future developments in chapter 8).

CHAPTER 6 - TQMS CONTROL SYSTEM -

The principal driving force behind the development of the distributed processing system in our laboratory was the need for an instrument control system that would allow the full potential of our Triple Quadrupole Mass Spectrometer (TQMS) to be utilized. The flexibility and complexity of this instrument requires a control system with capabilities that are greater than those that can be realistically implemented with a single processor control system. To assist in the discussion of this control system, the following section presents a description of the TQMS instrument and its principal operating modes.

THE TRIPLE QUADRUPOLE MASS SPECTROMETER

The triple quadrupole mass spectrometer is a type of tandem mass spectrometer that was invented in our laboratory by Yost and Enke(44),(45). The TQMS instrument is constructed from two quadrupole mass filters that are separated by a quadrupole collision chamber as shown is

Figure 6-1. In the TQMS technique, the sample to be analyzed is first ionized in the source region, and the resulting ions are mass analyzed in the first quadrupole mass filter. The selected ions are then fragmented in the collision chamber by collisions with other molecules. The collision chamber consists of an RF-only quadrupole which focuses the scattered ions thus increasing the efficiency of the collision process. The resulting fragment ions are then mass analyzed by the third quadrupole mass filter and detected by a continuous dynode electron multiplier.

The use of tandem mass spectrometers has proved to be of great value in the direct analysis of mixtures and the determination of organic structures. When the TQMS is used for mixture analysis, the sample is ionized using "soft ionization" which, results in predominantly molecular ions. These ions can then be selected by the first quadrupole, fragmented in the second quadrupole, and identified by the fragment spectra obtained by scanning the third quadrupole. For structure elucidation, the pure sample is ionized using "hard ionization", which results in fragmentation of the sample. Each of these parent ion fragments can be selected by the first quadrupole, fragmented in the second, and the resulting daughter ions analyzed by the third quadrupole. The compete fragmentation map is formed by recording all of the parent-daughter fragmentation steps; this map is a

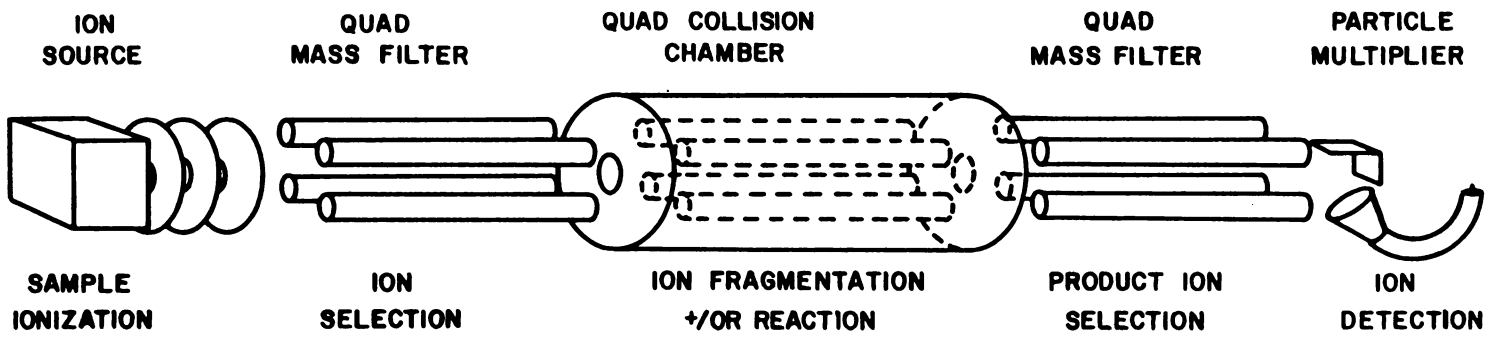

Figure 6-1 The TQMS Instrument.

powerful aid in determining the original structure of the sample.

The five basic operating modes of the TQMS instrument are shown in Figure 6-2. The first two of these modes, quad-one scan and quad-three scan, produce spectra that are similar to the spectra that are obtained from single quadrupole mass spectrometers. In these two modes the quad that is being scan is operated in a mass filter or DC mode, while the other quadrupoles are operated in a RF mode, which allows ions of all masses to pass. In the daughter scan mode, the first quad is set to select the desired parent ion from the source. After these ions are fragmented in the second quad, the third quad is scanned to produce a spectrum of all the daughter ions that result from the selected parent. In the parent scan mode the third quad is set to select a single daughter ion. The first quad is then scanned to select successive parent ions, which are then fragmented in quad two. This mode results in spectra that indicate all of the parent masses that fragment to give a specific daughter ion. The final mode of operation is a neutral loss scan. In this mode of operation, the first and third quads are both scanned at the same time, but at masses that differ by a constant amount. This mode results in the spectra of all parent ions that fragment and lose a neutral fragment which has a mass equal to the offset

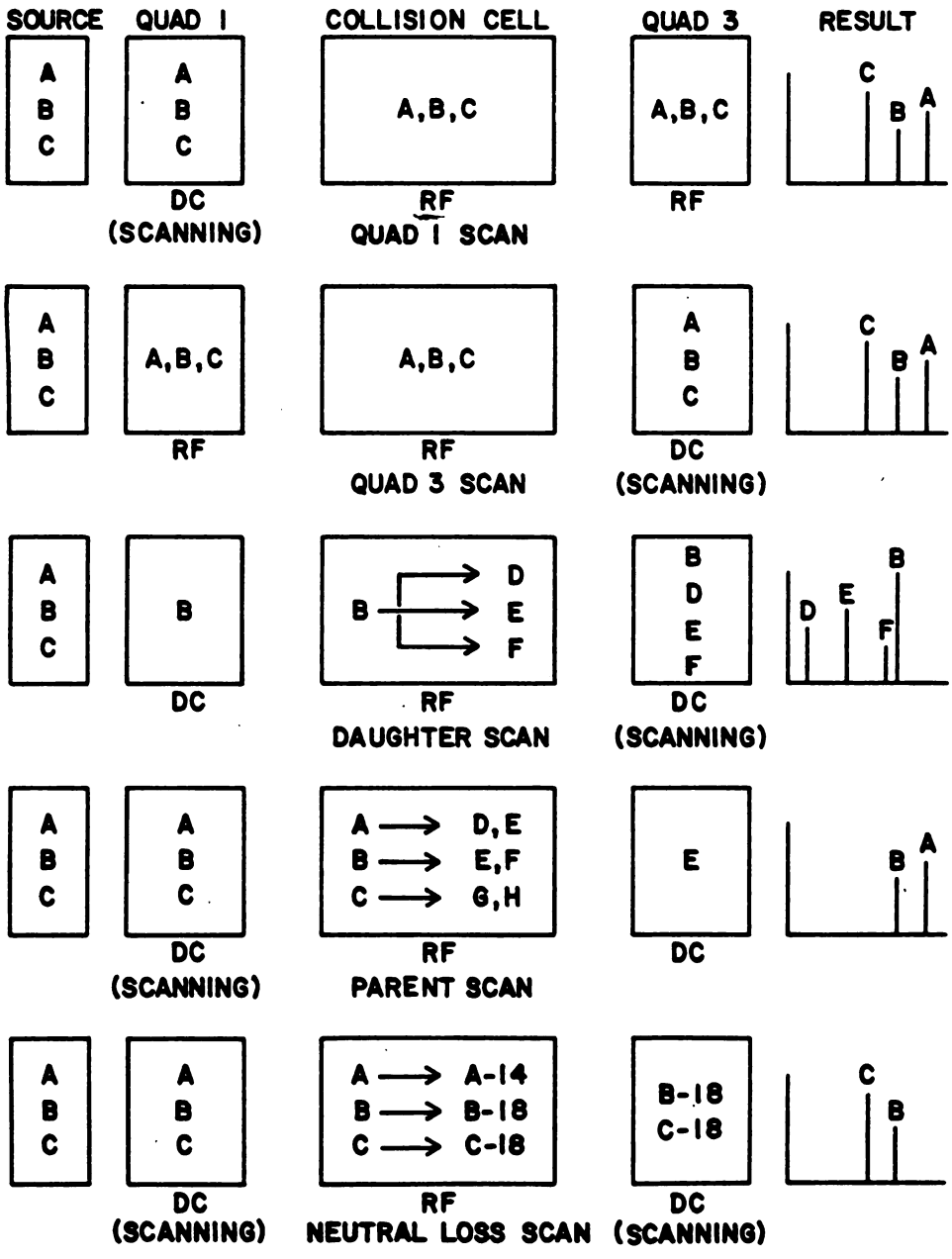

Figure 6-2 The TOMS Operating Modes.

between the quad-one mass and the quad-three mass.

REQUIREMENTS FOR A TQMS CONTROL SYSTEM

One of the characteristics of the TQMS that must be considered by the designer of the control system is the dynamic range of ion current. Because of the two stages of selectivity in the TQMS the ion current has a usable dynamic range of 10E8. To insure adequate time resolution when a GC or other transient sample introduction technique is used, it is desirable that the TQMS be able to scan at a rate of 1000 AMU/sec. At these scan rates, analog current measuring techniques are used, which results in a dynamic range of 10E5 to 10E6. This range is limited by ion statistics at the low end and by detector saturation at the high end. However, if a nontransient sample introduction technique is used, the ion signal can be measured over the full dynamic range by employing pulse counting techniques to measure the ion signal at the low end. Because of the time needed to accurately determine the ion signal pulse counting, this technique can only be used at slower scan rates. This wide dynamic range requires number representations and peak-finding algorithms that work with

wide ranges of data values.

The large number of devices that must be controlled on the TQMS also presents a challenge for the designer of the control system. These devices are listed in Table 6-1. The control system must also be capable of rapidly switching between the different ionization modes of the TQMS. The three principal ionization modes are electron impact ionization, positive chemical ionization, and negative chemical ionization. When the ionization mode is changed, the settings of all of the devices may have to be changed. This requires that the control system be capable of storing the settings of each device for every ionization mode. The values of several of the ion path devices should also change as the mass of the quadrupoles is scanned. This tracking of the ion path with mass greatly increases the number of calculations that the control system must be capable of performing in real time while scanning.

To enable the operator to interact with the instrument effectively, the control system needs the following capabilities: the ability to display the raw ion signal in real time, the ability to display the acquired data, utilities to manage the data that the system acquires, and a user friendly interface. A display of the raw ion signal is needed so the operator can optimize the different

Table 6-1. Devices in the TQMS to be controlled.

Electron energy
Repeller
EI ion volume
CI ion volume
Extractor
Ion source lens 1
Ion source lens 2
Ion source lens 3
Interquad lens 1-2
Interquad lens 2-3
Quad 1 DC rod offset
Quad 2 DC rod offset
Quad 3 DC rod offset
Multiplier high voltage
Mass selected by quad 1
Quad 1 delta mass
Quad 1 resolution
Mass selected by quad 2
Mass selected by quad 3
Quad 3 delta mass
Quad 3 resolution

instrument parameters. The display of the acquired data permits the operator to determine if the control system is collecting data properly and is also useful in monitoring the course of an experiment.

THE TQMS CONTROL SYSTEM

In the distributed control system that was built for the TQMS, the different tasks in the system were partitioned into separate processors using vertical partitioning. This resulted in the four-processor system, shown in Figure 6-3, with one master processor and three slave processors. The three slave processors perform the real time tasks with one processor handling the efferent tasks, a second the afferent tasks, and the third the central partition tasks. The master processor is a combined coordinator/communicator which also handles the communication with the operator, with the disk drives, and with a PDP 11/23, which acts as the data analysis system for the triple quadrupole mass spectrometer. The ion path slave processor performs the efferent tasks of controlling the quadrupole power supplies and the voltages on all of the ion path elements. The detection slave processor

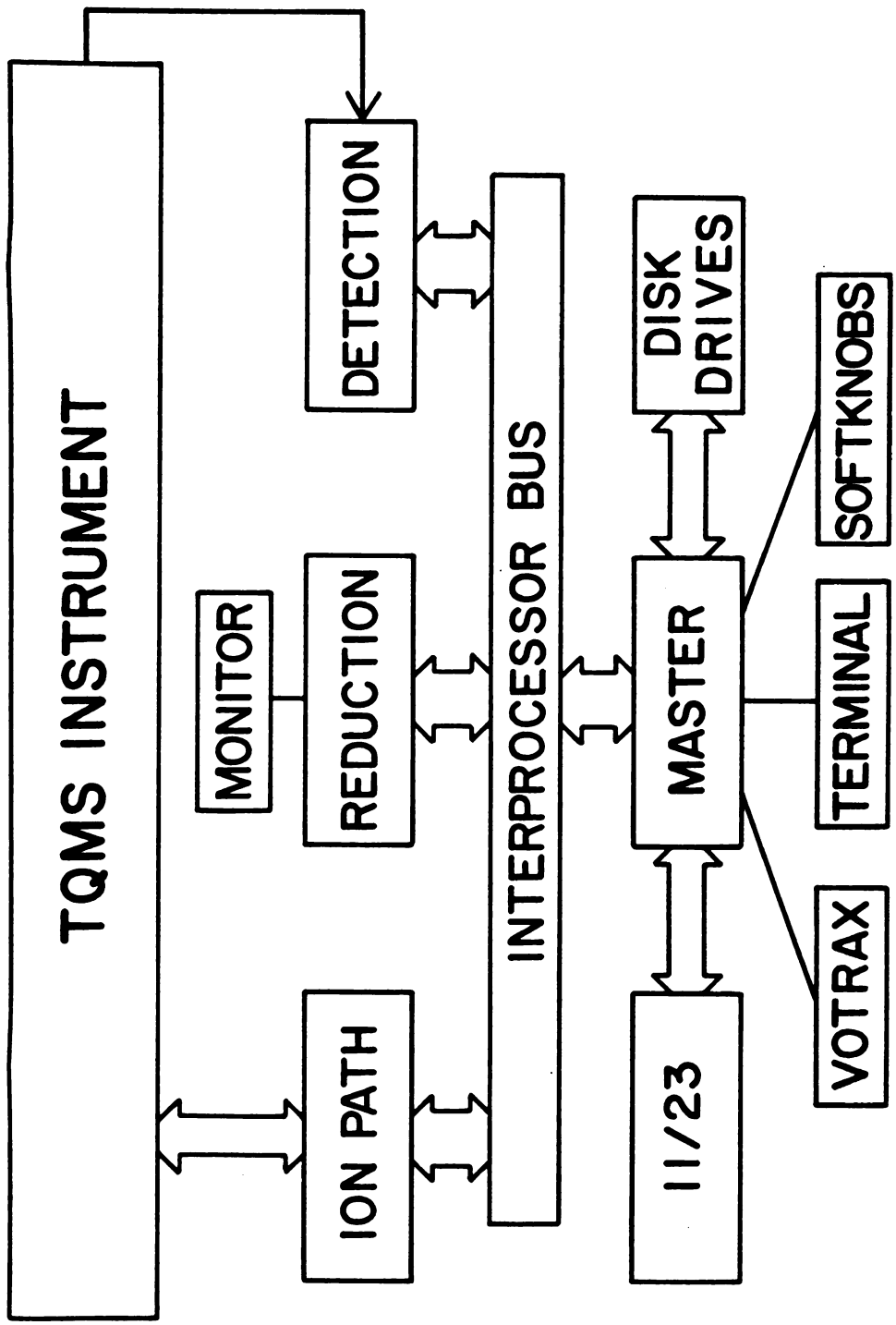

Figure 6-3 Block Diagram of the TQMS Control System.

performs the afferent tasks of ion signal conversion and data formatting. The central partition tasks are performed by the reduction slave processor, which performs the peak finding operations on the data from the other two slave processors.

MASTER PROCESSOR

A block diagram of the master processor is shown in Figure 6-4. This processor was built on five dual bus boards with the standard modules listed in Table 6-2. The master processor is a 5 MHz 8088 processor with an 8087 numeric coprocessor for increased "number-crunching" ability. The three RAM/ROM modules provide 8K of ROM and 40K of RAM memory for the system. The master processor has four serial ports that are used to communicate with a terminal, a printer, a Votrax voice synthesizer, and the PDP 11/23 data analysis system. A parallel I/O port is provided to transfer data to the 11/23 system while the serial port to the 11/23 is used for control purposes. An additional interface to the operator is provided by a set of softknobs. These optically encoded knobs are used to provided an enhanced user input. Mass storage for the distributed processing system is provided by an SCSI disk controller, which controls an 8 megabyte Winchester disk

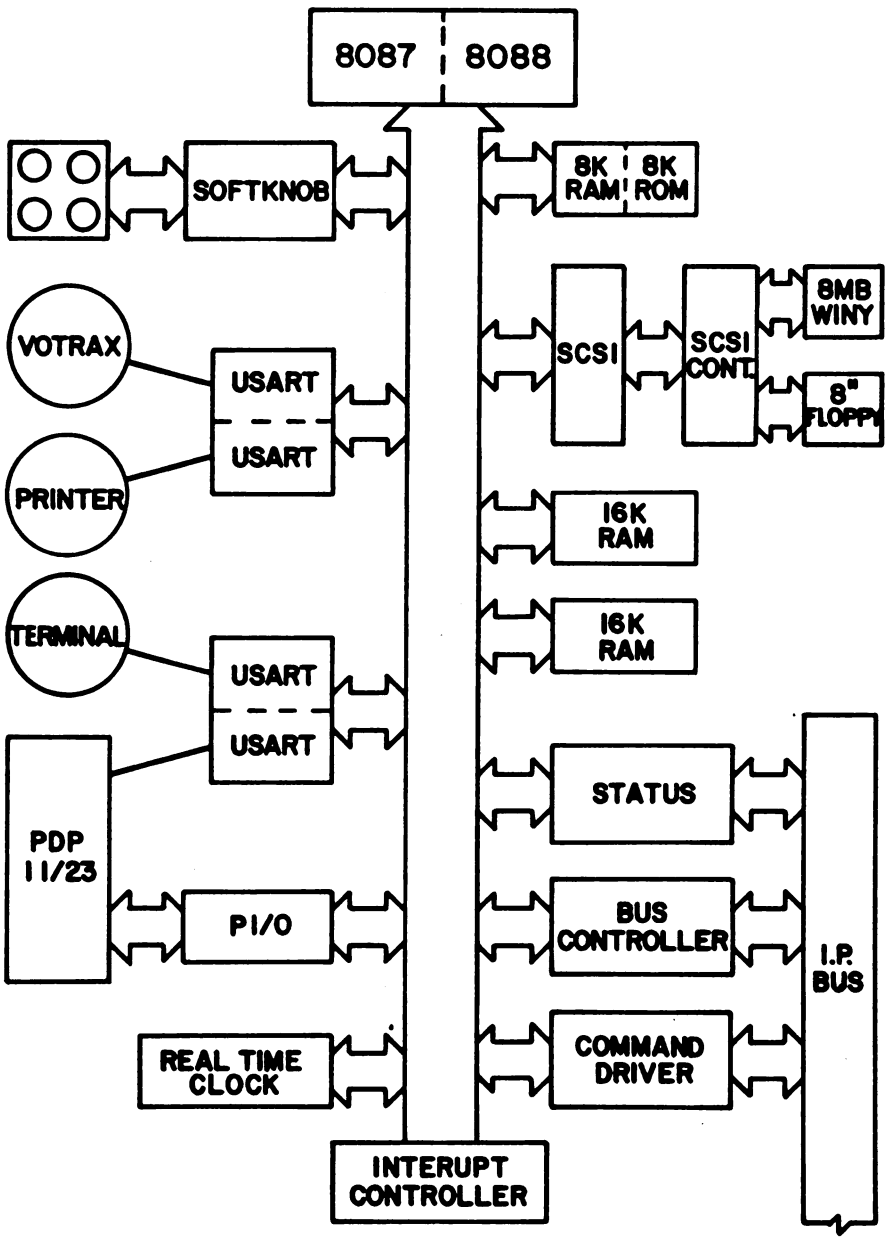

Figure 6-4 Block Diagram of the Master Processor.

Table 6-2. Component Modules of the Master Processor.

1 8088 CPU Module

1 8087 Numeric Coprocessor Module

1 Interrupt Controller Module

3 RAM/ROM Modules

3 Chip Select Modules

4 Address Extender Modules

2 AC Terminator Modules

2 Dual USART Modules

1 Parallel I/O Module

1 SCSI Adapter Module

1 Real Time Clock Module

1 Softknob Interface Module

1 Interprocessor Status Module

1 Interprocessor Bus Controller Module

1 Interprocessor Command Driver Module

and an 8" floppy disk. The master processor is also provided with a real time clock which allows it to enter the time and date into all experimental records. This processor is a combined coordinator/communicator so it has an interprocessor bus controller, an interprocessor command driver, and an interprocessor status module.

ION PATH PROCESSOR

The ion path slave processor, which is shown in Figure 6-5, is built from the modules listed in Table 6-3. This processor is an 8 MHz 8088 processor that is supplied with 16K bytes of RAM memory. Since this is a slave processor, the interprocessor modules consist of a status module, a bus switch module, and a command buffer module. The ion source electronics are controlled by an octal DAC module and part of a parallel I/O module. Another port on the parallel I/O module is used to control the vertical gain on an oscilloscope display. A differential transceiver module is used to communicate with a remote RF control box which houses the DACs that are used to control the three quadrupole power supplies and the interquad lenses. There are two special modules on this processor; one is the AMU timer module which is used to determine the scan rate of the quadrupoles, and the other is a link/sync output module

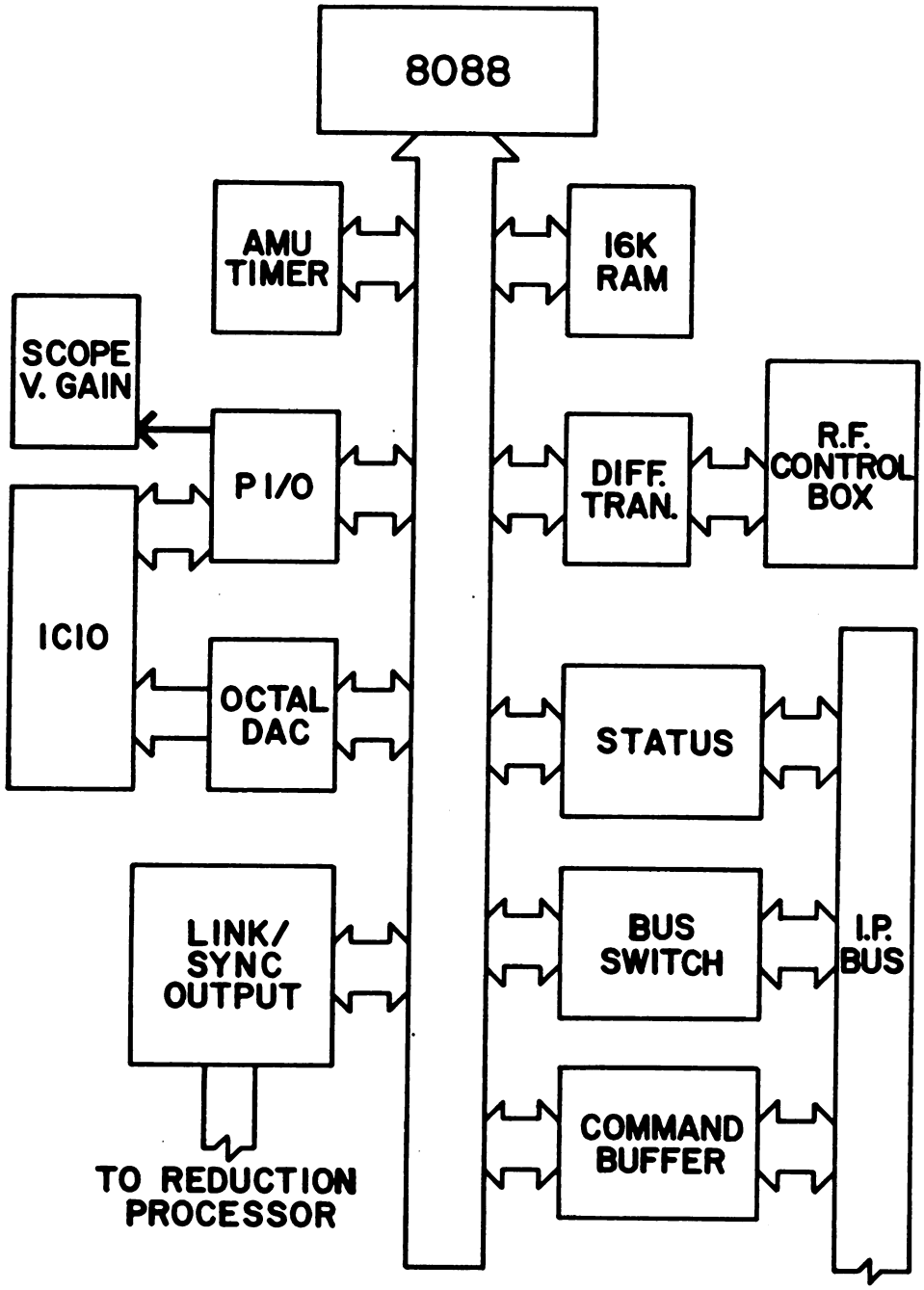

Figure 6-5 Block Diagram of the Ion Path Processor.

Table 6-3. Modules in the Ion Path Processor.

1 8088 CPU Module

1 RAM/ROM Module

1 AC Terminator Module

1 Parallel I/O Module

1 Octal DAC Module

1 Differential Transceiver Module

1 AMU Timer Module

1 Link/Sync Output Module

1 Interprocessor Status Module

1 Interprocessor Bus Switch Module

1 Interprocessor Command Buffer Module

which is used to transfer data to the reduction processor and to synchronize the data acquisition process. This board could be eliminated by a redesign of the status module using recently available integrated circuits.

DETECTION PROCESSOR

Figure 6-6 shows a block diagram of the detection slave processor, which is built from the modules listed in Table 6-4. This processor uses an 8 MHz 8088 CPU and has 16 kbytes of RAM memory. The ion current signal is converted to a voltage signal by a preamp module, which has a transresistance of 2 megohms. This results in a full scale 10 volt output for an input current of 5 microamps, which is the maximum current that should be drawn from the continuous dynode electron multiplier. The signal from the preamp is then sent to a multiamp module which provides five amplifiers with gains of 1, 4, 16, 64, and 256. These five signals are connected to a differential multiplexer module on the detection processor which is used to select one of the channels for conversion by the ADC. The multiamp module also contains a set of comparators and digital logic that produces a digital code to indicate the proper amplifier to use. This code is read by the processor using a parallel I/O port and is then used to select the

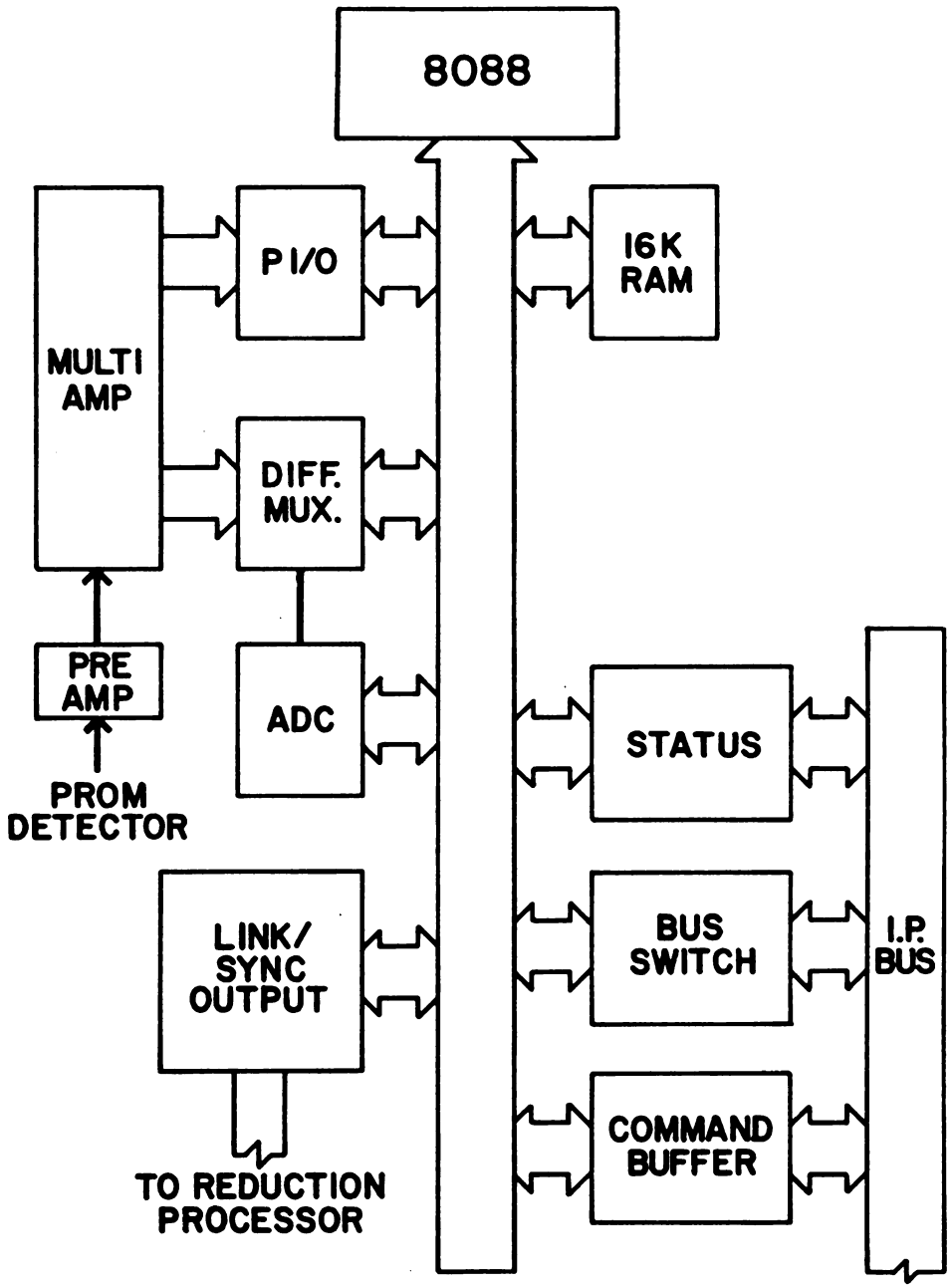

Figure 6-6 Block Diagram of the Detection Processor.

Table 6-4. Modules in the Detection Processor.

1	8088 CPU Module
1	RAM/ROM Module
1	AC Terminator Module
1	Parallel I/O Module
1	Differential Multiplexer Module
1	12-Bit ADC Module
1	Link/Sync Output Module
1	Interprocessor Status Module
1	Interprocessor Bus Switch Module
1	Interprocessor Command Buffer Module

differential multiplexer channel. The data from the ADC are formatted and sent to the reduction processor via a link/sync output module. Although this processor is currently under-utilized by just operating the ADC, it provides the processing power needed to operate an additional pulse counting interface, which will be provided in the future.

REDUCTION PROCESSOR

The reduction slave processor, which is shown in Figure 6-7, is constructed from the modules listed in Table 6-5. The reduction processor is a 5 MHz 8088 processor with an 8087 numeric coprocessor, which provides additional "number-crunching" capabilities. Two RAM/ROM modules are used to provide 32 kbytes of RAM memory for the processor. This processor receives data from the ion path processor and the detection processor through two link/sync input modules and then performs peak-finding operations in real time. This processor is also equipped with a graphics controller module and a graphics memory plane populated with 64K-bit DRAMs. These two modules provide a 1024 by 780 pixel graphics display that is used to display the acquired data.

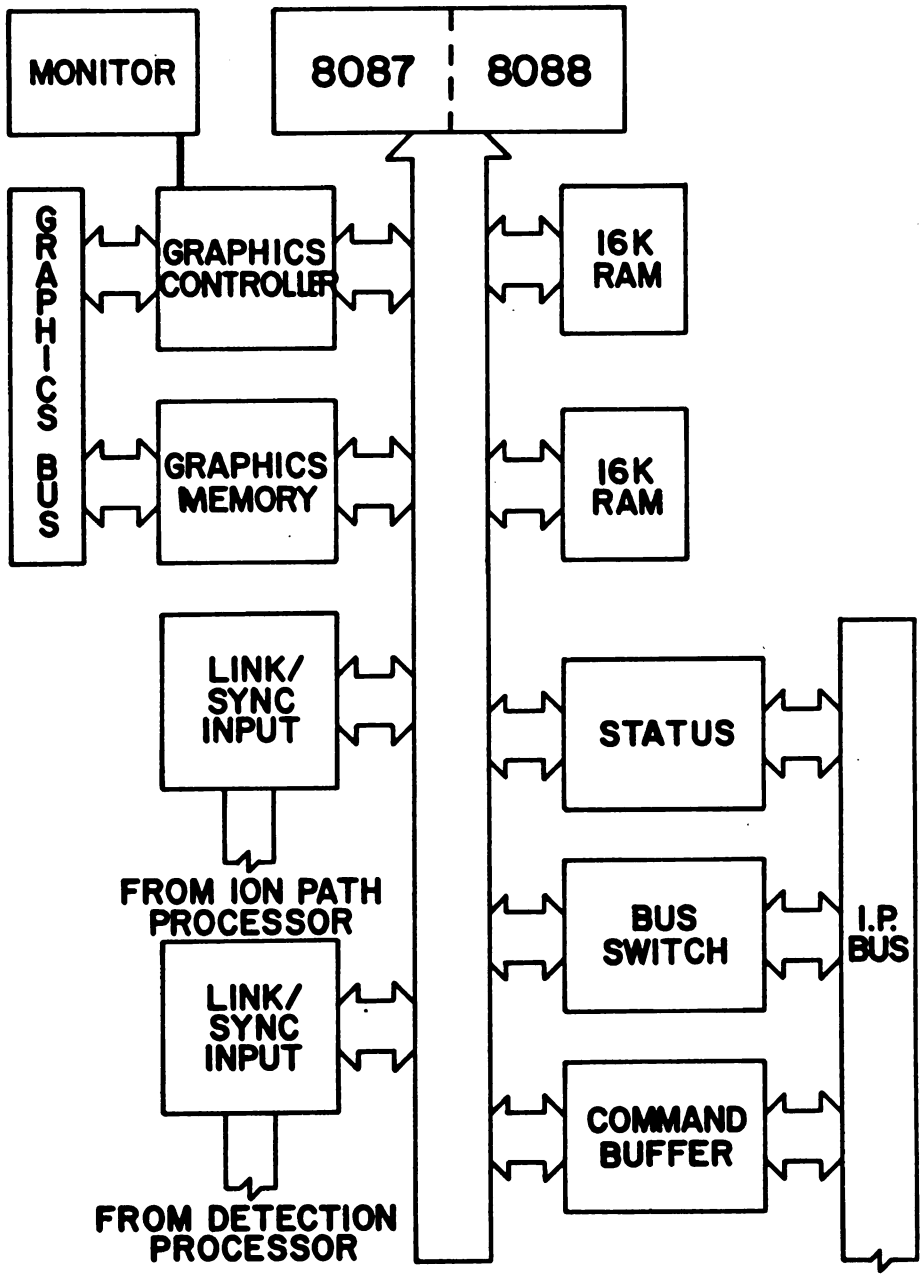

Figure 6-7 Block Diagram of the Reduction Processor.

Table 6-5. Modules in the Reduction Processor.

1	8088 CPU Module
1	8087 Numeric Coprocessor Module
2	RAM/ROM Modules
1	Graphics Controller Module
1	Graphics Memory Plane Module
2	Link/Sync Input Modules
1	Interprocessor Status Module
1	Interprocessor Bus Switch Module
1	Interprocessor Command Buffer Module

DATA ACQUISITION

The interaction of the three slave processors during data acquisition is illustrated in Figure 6-8. These processors execute a set of tightly coupled tasks to acquire and reduce a scan of data from the TQMS. This data record is then transferred to the master processor where it is stored on the disk while the three slave processors are acquiring the next scan of data. To initiate a scan operation, the master processor transfers command and parameter information to each of the three slaves. The ion path processor then calculates and sets the first set of ion path values. When these values are stable, the ion path slave passes the X-axis data to the reduction processor, informs the detection processor that it can acquire a data point, and begins to calculate the next set of ion path values. The detection processor then acquires the data point, informs the ion path slave that it can update the ion path values, formats the acquired data point, and checks to see if the reduction processor is ready to accept the new data. When the reduction processor is ready, the detection processor passes it the Y-axis data and waits until the ion path processor indicates that the next data point should be acquired. At the same time the reduction

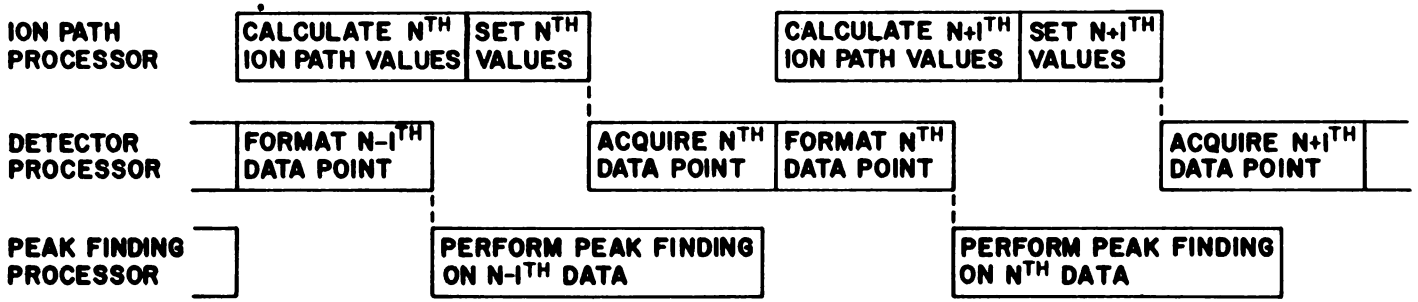

Figure 6-8 Diagram of TQMS Data Acquisition.

slave is performing peak finding using the X and Y axis values that were passed to it by the other two processors. This process continues until all of the data points in the scan have been acquired and processed. While all of this is taking place, the master processor is storing the data from the previous scan and is interacting with the operator to determine the next operation to perform.

USER INTERFACE

The TQMS control system can interact with the operator through several different devices. These devices include a CRT terminal, a printer, a graphics monitor, a set of softknobs, and a speech synthesizer, which combine to form an efficient and user friendly interface to the system. Because of the many instrument parameters and operating modes, special attention was focused on the interaction between the operator and the control system. This attention resulted in a set of simple descriptive commands and several interactive screen editors that can be used to modify readily any of the instrument parameters(46),(47). The control system also provides several aids for the optimization of the different instrument parameters. One of

these is the ability to display the raw ion signal on an oscilloscope using either a fast scan command, which scans over a range of masses, or a split screen command, which allows up to five selected mass windows to be displayed next to each other. During these displays the various instrument parameters can be modified by using a set of softknobs. These softknobs are rotary encoders which provide the feel of conventional potentiometers, but have the added advantages of variable resolution, unlimited range of rotation, and computer controlled starting value and clipping to preset limits. The control system is equipped with four of these encoders, which can be assigned to any parameter in the system.

A graphics display is provided so that the acquired data can be displayed to determine if the proper instrument parameters were used for data acquisition. The display also permits the course of an experiment to be monitored. A final means of interaction between the operator and the instrument is provided by a Votrax Type-n-Talk speech synthesizer(48). This module is used to provided a variety of warning and error messages, which permits the operator to move around the room and still be apprised of any problems with the TQMS instrument.

ENHANCED PERFORMANCE AND CAPABILITIES

This control system has several advantages over a similar control system that was implemented using only a single 8 MHz 8086 processor. The first of these is an increased scan speed which results in better time resolution when a GC or similar sample introduction technique is used. The scanning rates and the number of data values that are averaged for each data point are shown for the two systems in Table 6-6. As can be seen, the distributed control system is capable of scanning at rates of 1000 and 500 AMU per second which the single processor system cannot perform. At the faster scan rates, which are the more commonly used rates, the distributed system has twice the signal-to-noise ratio of the single processor system. The distributed system is also capable of tracking several of the ion path devices with mass during a scan operation, which improves the resolution and sensitivity of the instrument. The distributed system should be capable of performing this tracking operation even at the fastest scan rates since the limiting factor in the scan rates is the ability of the detection processor to acquire and format a data point.

Table 6-6. Scanning Rates for the TQMS Control System.

AMU/SEC (10 points/AMU)	#AVERAGES/PNT single processor	#AVERAGES/PNT distrib. processor
1000	---	1
500	---	2
250	1	4
100	4	16
50	16	32
25	32	64
10	128	128
5	256	256
2.5	512	512
1	1024	1024

The increased processing power of this system will also permit new features to be developed such as the use of pulse counting for the detection of low level ion currents and the implementation of intelligent scanning algorithms which scan rapidly where no ion current is found and scan slowly when a peak is found. These features will increase the sensitivity of the instrument and reduce the amount of sample needed for an analysis. Another example of the advantages of separating the tasks into different processors is the use of the softknobs during a split screen or fast scan display. In the single processor system the softknobs task slows the scan display by over 40% which causes distortion in the display. In the distributed processor system the softknob task runs on a separate processor from the scan tasks and this results in less than a 5% change in the execution time of the scan tasks when the softknobs are running.

The success of the TQMS control system can be further illustrated by discussing the advantages of distributed processing systems that are listed in Table 4-1. The first of the advantages is parallel execution which is demonstrated in the TQMS system during the data collection stage. In the present system the bottleneck in the collection process is the detection processor. However if the present ADC was replaced with a faster ADC the

bottleneck would shift to the reduction processor. The bottleneck could even be shifted from the reduction processor to the ion path processor if some specialized signal processing hardware was added to the reduction processor. The fact that the bottleneck in the data acquisition operation can be moved between the different processors indicates that each processor is executing a major portion of the overall task in parallel with the other processors. The experience with this system is that the "overhead" consumes between 0% and 20% of the processor time depending on the specific task. This is substantially less than the overhead fraction in the single processor system. The simpler bus structure and the lower complexity of the different modules also allows the easier addition of new interfaces and capabilities to the system as compared to the single Multibus processor system.

The advantages of non-interference of tasks is demonstrated by the Softknobs task mentioned above. The separation of the ion path control tasks and the softknobs driver tasks onto different processors resulted in superior performance because the interference between the two tasks was eliminated. Also eliminated were the task interleaving programs and the priority assignment programs. The simpler modification of tasks can be shown by examining the data acquisition operation. In the distributed processing system

the tasks executing on the reduction processor, for example, could be changed or modified without affecting the tasks running on either the ion path or the detection processors. In the single processor system however, the three tasks are tightly interleaved so any change in one routine can easily affect the other tasks.

The modulatity of the hardware and the software in the distributed processing system also greatly facilitates the enhancement of the instrument. If more processing power is needed to add a new feature to the instrument another processor can be added without major changes to the system while in the single processor system adding more processing power means starting over with a new processor. The distributed processing system is also easier to debug and troubleshoot because of the ease with which a problem can be confined to a section of the overall system and also because of the capabilities of the Master processor to examine and change any of the slaves memory or peripherals. As these examples show, the increased processing power of the distributed system permits greatly enhanced control of the TQMS. In the future as the control system is further developed the advantages of the distributed processing system will become even greater.

CHAPTER 7 - SOFTWARE -

Although the software for the various systems described in the previous chapters is not part of the research that this dissertation covers, this chapter will provide a short discussion of the software that was written for these systems. This discussion will focus on the language that was employed, the modifications that were made to it for use in the distributed processing environment, and the resulting commands for the TQMS control system.

FORTH LANGUAGE

The software for the different control systems was implemented using the polyFORTH version of the FORTH language(49). This version of the FORTH language has several attributes that make it an excellent language for instrument control systems. Some of these advantages are its size, the ability to directly access machine resources, its execution speed, and its extensibility. A complete

FORTH system that contains the FORTH compiler, an editor, an assembler, and disk and terminal handlers occupies only 8 kbytes of memory. This standard FORTH system contains all of the functions needed for program development and therefore does not require the use of cross compilers or separate development systems. The standard FORTH system also allows direct access to any memory address or I/O port without the intervening presence of an operating system. This is extremely advantageous in real time control systems where much of the software involves interaction with the instrument interfaces. This implementation of FORTH can produce high-level threaded code that executes at 30% to 50% of the speed of a similar routine written in assembly language. This is an excellent speed considering the advantages of writing in a high level interpretive environment. However, if a routine needs maximum speed, the FORTH system also provides an assembler as an intergal part of the standard system.

One of the most useful attributes of FORTH is its extensibility by which new commands and structures can be readily added(50). In FORTH a new program or "word" is defined using a series of previously defined words. This allows new programs to be easily written by merely concatenating existing words. Parameters are transferred between the different words by the use of a stack which

results in an identical interface for both high level and assembly routines. The end result of the process of defining new words from previous words is a high level language that is specific to the application. This not only permits efficient operation of the instrument but also provides a natural path by which complete experiments can be programed.

DISTRIBUTED FORTH

For operation in the distributed processing environment several changes were made to the basic FORTH system(51). Extensions were made to the FORTH system on the master processor so that it could download code to the slaves, perform interprocessor transfer operations, and direct the execution of tasks on the slaves. First a set of words were written to control the different interprocessor communication modes. These words were then used to modify the target compiler, which is normally used to create new ROMable FORTH systems, so that the compiled code was directly written into the slave's memory instead of being stored on disk. The target compiler was also modified so that it created a Slave Command Access Table (SCAT) which

contained the information necessary to direct the execution of a task in the slave processor.

In the versions of FORTH for the slave processors, the editor, assembler, and disk routines were removed since they would not be needed and the standard command interpreter was modified to accept commands and parameters from the FIFO buffers on the command buffer module. This was accomplished by using four of the bits in the high order byte of the command to encode the operation that is to be performed. If these bits have a value of one the command is an immediate operation and the rest of the command is discarded. If the bits have a value of two, the lower sixteen bits of the command are the address of a FORTH word to be executed; if the value of the bits is three, the lower sixteen bits of the command are a parameter to be pushed onto the stack. The end result of these modification was an integrated system where all of the processors are programmed in a common language that deviates little from a standard FORTH system.

TQMS SOFTWARE

As examples of the type of high level commands that result from the use of FORTH, some of the commands from the TQMS control system are presented in Table 7-1(47). These commands are high level mass spec commands that allow an operator to use the instrument by typing them in either singly or several on a line. However, these same commands can be used to create new words that assist the operator in performing experiments. The operator has little sense that they are really programing a multiple processor system. By the creation of such software the power and performance of a complex distributed processing system can be easily controlled by even a novice user.

SUMMARY

The power of the FORTH language at implementing such systems is illustrated by the fact that the modification to FORTH for operation in the distributed environment amounted to only about 15 blocks of code out of the several thousand blocks of code that make up the control system software.

Table 7-1. Selected TQMS Control System Commands

COMMAND	FUNCTION
1SCAN	Quad One Scan
3SCAN	Quad Three Scan
PSCAN	Parent Scan
DSCAN	Daughter Scan
NSCAN	Neutral Loss Scan
DISP	Plot Acquired Data
DLIST	List Acquired Data Points
KNOBS	Activate Softknobs
PED	Activate Parameter Editor
SPLITS	Setup Split Screen Display
SET	Set a Selected Device to a Value
MS1	Set Quad One to a Specified Mass
MS3	Set Quad Three to a Specified Mass
ADD	Add Two Spectra Together
SUB	Subtract Two Spectra
SDIR	Display Directory of Acquired Scans
EI	Select EI Ionization Mode
+CI	Select Positive Chemical Ionization Mode
-CI	Select Negative Chemical Ionization Mode
HELP	Display Help Information

The modular nature of FORTH also combined with the modular hardware to create a system that was readily tailored to the need of the TQMS instrument. The creation of this system also provides a firm foundation for the creation of truly intelligent control systems that will be able to intelligently acquire data and optimize the instrument in real time during data acquisition. This optimization will be performed using rules that are derived from the experimental goals and will enable the control system to extract the best information from the available sample.

CHAPTER 8 - FUTURE DEVELOPMENTS -

The principal short range goals in the area of instrument control systems should be the consolidation of gains already made and the wider dissemination of information on the modules that are currently available. For example several modules that were designed for the TQMS, such as the softknobs and the graphics modules, have yet to be applied to other instruments where they could be very useful. During the development of this microprocessor system, a great deal of experience with the design of microprocessor systems was acquired. This experience, combined with developments in the electronics industry in recent years, resulted in a number of ideas that would be interesting to investigate in the future. These ideas can be grouped into three areas: new modules for the single microprocessor system, an improved implementation of the interprocessor modules, and ideas for the future development of the TQMS control system.

IDEAS FOR THE SINGLE PROCESSOR SYSTEM

At present the types of standard peripheral modules that could use the most development are the various analog functions and the converter modules. Both the design of some existing modules and the variety of available modules could be improved. Another peripheral module that could be investigated is the use of a National MM54240 remote controller chip for interfacing to different types of ancillary equipment. This module provides a simple two wire bus that can connect up to 64 remote modules to the master module thus permitting an inexpensive method of connecting the computer system to different instruments.

The design of several new processor boards could also be investigated for their suitability for use in instrument control systems. The first of these would be a CPU module that was implemented using the NS08032, an 8-bit version of the National 16000 series microprocessor. This processor has a large contiguous memory space and a completely orthogonal instruction set which results in easier software development. This processor is also capable of supporting user-designed custom coprocessors which would be an interesting area to examine. The control signals on this

processor are similar to those employed by Intel processors which would simplify the task of designing the module.

A second processor that would be interesting to implement would be one that is based on the NCR/32 VLSI chip set. This chip set is a user-microprogrammable 32-bit microprocessor. A 32-bit version of polyForth is available for this processor which is capable of executing simple 32-bit FORTH instructions in 1.2 to 4 microseconds and has full support for user microcoding. This performance level is greater than microprogrammed FORTH on a VAX 780. A possible approach to implementing this processor would be the design of a processor and memory board that is the same size as a dual bus board. This would allow the bus width between the processor and memory to be 32-bits wide while still allowing the existing peripherals to be accessed using the local bus.

Another module that would be useful for signal processing applications would be a peripheral module that utilizes the TI TMS32010 digital signal processor. This processor and its memory could be implemented on a single memory mapped-module which would allow the system processor to load code into the memory. This module would greatly increase the amount of signal processing that could be performed in real time by the microprocessor systems.

Examples of the type of operation that this module could perform are centroiding calculations for peak-finding, Fourier transform calculations, and digital filtering operations.

INTERPROCESSOR HARDWARE MODULES

The implementation of the TQMS control system demonstrates that while the concepts of the different interprocessor communication paths were sound, the implementation of these modules could be improved. The modules that support the three different paths will be examined in turn. The first path to be examined is the direct memory transfer path. The main change that should be made to the bus controller and the bus switch is the replacement of the control logic on both modules with PROM or PAL based state machines. This would reduce the complexity of the circuit and also increase the reliability of the transfer operation. A second change that could be examined is the use of National DS3662 trapezoidal bus transceivers instead of the 74LS drivers that are currently employed. Their use should result in increased noise immunity of the system.

The second path is the command transfer path that is implemented by the command driver and command buffer modules. Again the control logic on these modules should be replaced with PROM or PAL based state machines. Several newly introduced FIFO buffers should also be examined to determine if they have any advantages over the FIFOs that are currently being used.

The status transfer path needs the most extensive modifications. These consist of increasing the amount of dual-ported memory on each status module and the addition of a new mode of status transfer. Synertek has recently introduced an SY2130 dual-ported RAM chip which is organized as a 1024 by 8 bit memory. The use of this chip would allow each processor in the system to have 127 bytes of software status and one byte of hardware status. This would greatly ease the transfer of small amounts of information between different processors. The new mode of status transfer that needs to be implemented is a flag transfer mode which would allow a processor to set a flip/flop on any of the other processors in the system. The output of the flip/flop could be used for an interrupt or TEST\ input to the receiving processor which could then clear the flip/flop. The implementation of this mode would greatly reduce the handshaking that is needed to sychnronize tasks on different processors. The status

transfer processes could be implemented as a four cycle operation for each processor. The four cycles would be: the transfer of the flags, the transfer of the hardware status, the transfer of the software status address, and the transfer of the software status data. With a 100 ns cycle time, the status information on all the processors in an eight-processor system could be updated in 3.2 microseconds. This module should also be implemented with state machine control logic and trapezoidal bus drivers.

TQMS CONTROL SYSTEM

To improve the data acquisition ability of the TQMS control system, the hardware for pulse counting should be installed along with a faster ADC. This would enable the measurement of low level ion signals and would permit the detection processor to perform more signal averaging for a better signal-to-noise ratio. The addition of a fifth 8088 slave processor which would control the graphic monitor would allow the data to be displayed in real time as an experiment is running. The addition of signal processing hardware, such as the TI TMS32010, would increase the signal processing capabilities of the reduction processor

and allow the use of more complex peak finding algorithms. Finally, the control system could be interfaced to the various ancillary equipment that is connected to the TQMS.

These are just some suggestions of ideas that would be interesting to investigate. The needs of future instrument control systems will also provide a great number of other ideas.

And they all lived happily ever after.

-THE END-

REFERENCES

REFERENCES

(1) Enke, C.G., Science, 1982, 215, 785.

(2) Kalivas, J.H., Kowalski, B.R., Anal. Chem. 1982, 54, 560.

(3) Overton,M.W., Alber,L.L., and Smith,D.E., Anal. Chem., 1975, 47, 363A.

(4) DeVoe,J.R., Shideler,R.W., Ruegg,F.C., Aronson,J.P., and Shoenfeld,P.S., Anal. Chem. 1974, 46, 509.

(5) Ziegler,E., Henneberg,P., and Schamburg,G., Anal. Chem. 1970, 42, 51A.

(6)Perrin,D.D., Talanta, 1977, 24, 339.

(7) Betterigde,D., and Goad,T.B., Analyst, 1981, 106, 257.

(8)Schmidt, M., Rev. Sci. Inst. 1983, 54, 1446.

(9)Caceci, M.S., Choppin, G.R., Comp. and Chem. 1982, 6, 161.

(10)Vickers, G.H., Trask, T.O., Parli, J.D., Wisman, J.A., Durham, B., Green, R.B., Chem., Biomed., and Environ. Instru., 1982, 12, 289.

(11) Avdeef,A., and Bucher,J.J., Anal. Chem. 1978, 50, 587.

(12) Lindsay, W.S., Kizzort, B.L., Justice, J.B., Salamone, J.D., Neill, D.B., Chem., Biomed., and Environ. Instru., 1980, 10, 311.

(13)Woodward, W.S., and Reilley, C.N., Pure Appl. Chem., 1978,50,785.

(14) Wu,A.H.B., and Malmstadt, H.V., Anal. Chem., 1978, 50, 2090.

(15) Dessy, R.E., Anal. Chim. Acta., 1978, 103, 459.

(16) Dessy, R.E., Anal. Chem.1982, 54, 1167A.

(17) He,P., Avery,J.P., Faulkner,L.R., Anal. Chem. 1982, 54. 1313A.

(18) Avery,J.P., Diss. Abstr. Int. B. 1978, 39, 2253.

(19) Lovse, D.W., Diss. Abstr. Int. B, 1978, 38, 4771.

(20) Carlson, E., PhD Thesis, Michigan State University, E.Lansing, MI.

(21) Hornshuh, J., PhD thesis, Michigan State University, E.Lansing, MI.

(22) Intel 8085 Users Manual, Intel Corp., Ca.

(23) Intel 8086 and 8088 Users Manual, Intel Corp., Ca.

(24) Thiim, R., Master Thesis, Michigan State University, E.Lansing, MI

(25) Ratzlaff, E.H., PhD Thesis, Michigan State University, E.Lansing, MI.

(26) Jones,L.M., PhD Thesis, Michigan State University, E.Lansing, MI.

(27) Stanley, J., Masters Thesis, Michigan State University, E.Lansing, MI.

(28)Penix, N., Private Communication.

(29) Jones, L.M. Leroi, G.E., Myerholtz, C.A., Enke, C.G., Rev. Sci. Inst. Accepted for publication 10/83.

(30)Aiello, P.J., Enke, C.G, ACS Symposium "Image Devices in Spectroscopy", in press

(31) Aiello, P.J., Doctoral Thesis 1983, Michigan State University, E.Lansing, MI.

(32)Stults, J.T, Newcome, B.H., Myerholtz, C.A., Enke, C.G., 31st Annual ASMS conference, Boston, MA, 1983

(33) Stults, J.T., Enke, C.G., and Holland, J.F., Anal. Chem. 1983, 55 ,1323.

(34)Enke, C.G., Proc. 28th IUPAC Conference, Vancouver, BC (1981)

(35)Fathi, E.T., Krieder, M. Computer, March 1983, P. 23

(36) Searle,B.C., Freberg,D.E., Computer, 22. Oct. 1975,

(37) Lawson, J.T., Mariani, M.P., Proceedings of the IEEE, 358 (1978)

(38)Yourdon, E.,Constantine L.L., "Structured Design", Yourdon Incl., 1133 Ave. of the Americas, New York, NY, February 1976.

(39) Anderson, G.A., and Jensen, E.D., Computing Surveys, 1975, 7(4), 197

(40) HP-IL is a trade mark of Hewlet-Packard, Palo Alto, CA.

(41) IBM, IBM system/360 and system/370 Attached Support Processor Version 3 Asymmetrical Multiprocessor System:

general information manual GH20-1173.

(42) IEEE standard 488-1978

(43) MIL. STD. 1553 avionics bus

(44) Yost, R.A., Enke, C.G., J. Am. Chem. Soc., 100, 2274, (1978).

(45) Yost, R.A., Enke, C.G., Anal. Chem., 51, 1251A, (1979).

(46) Myerholtz, C.A., PhD Thesis, Michigan State University, E.Lansing, Mi.

(47) Myerholtz, C.A., Newcome, B.H., Enke, C.G., Submitted to Anal. Chem..

(48) Type-n-Talk is a trade mark of the Votrax division, Federal Screw Works, Troy, MI.

(49) polyFORTH is a trademark of FORTH, inc., Hermosa Beach, CA.

(50) Brodie, L., "Starting FORTH", Prentice Hall, Englewood Cliffs, NJ, 1981.

(51) Myerholtz, C.A., and Enke, C.G., Rev. Sci. Inst., submitted 1983.