MINICOMPUTER BASED MICROPROCESSOR DEVELOPMENT SYSTEMS

Richard Moses

A Thesis
in
The Department
of
Electrical Engineering

Presented in Partial Fulfillment of the Requirements
for the degree of Master of Engineering
at Concordia University
Montreal, Quebec, Canada

August, 1979

ABSTRACT

MINICOMPUTER BASED
MICROPROCESSOR DEVELOPMENT SYSTEMS

Richard Moses

Tailoring ones own development system for microprocessor applications provides an alternative to purchasing an existing system. With the aid of a minicomputer,a system may be configured to the requirments of the user while staying in the cost boundaries of what would be required for investing in an off-the-shelf system. The main feature is that the system is adaptable to any microprocessor family and may be upgraded as technology advances.

Such a system has been developed at the microprocessor applications laboratory as part of the Department of Electrical Engineering at Concordia University. Configured around a DEC PDP 11/45 minicomputer the system has been applied to the RCA CDP1802 microprocessor.

The 1802 microprocessor along with 1Kbytes of

memory is incorporated into a single board microcomputer module. During prototype development the microcomputer module is linked through an extender cable to the microprocessor development system via a custom designed hardware interface. Using RT-11 software, system and application programs written in RCA microprocessor assembly language are editted, assembled linked and executed by the PDP 11/45 minicomputer.

ACKNOWLEDGEMENTS

The author acknowledges his indebtedness to J.Kishon and R.Beyar for their kindness in offering their opinions and technical assistance throughout the research development. In addition, acknowlegement is due to to S.A.Gracovetsky whose influence as an educator and a leader will always be remembered, as well as V.Ramachandran who contributed time and ideas for the generation of the thesis. The research described here was supported in part by the National Research Council of Canada under contract number 1560. Finally credit is also due to AES Data.Ltd. for supplying the facilities and materials for the preparation of the manuscript.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

SYMBOLS AND ABREVEATIONS

ICE	In Circuit Emulation
PROM	Programable Read Only Memory
RAM	Random Access Memory
DEC	Digital Equipment Corporation
I/O	Input/Output
PC	Printed Circuit
CPU	Central Processing Unit
V	Volts
A	Amps
CMOS	Complementary Metal Oxide Semiconductor
IC	Integrated Circuit
SOS	Silicon On Saphire
MHz	Megahertz
Kbytes	1024 bytes
us	microseconds
ns	nanoseconds
R/W	Read/Write
MSB	Most Significant Bit

DMA	Direct Memory Access
TTL	Transistor Transistor Logic
no-op	no operation
LSI	Large Scale Integration
PPI	Programable Peripheral Interface
CRT	Cathode Ray Tube
A/D	Analog to Digital
D/A	Digital to Analog

CHAPTER 1

THE NEED FOR MICROPROCESSOR DEVELOPMENT TOOLS

1.1 INTRODUCTION

The complexity of systems have dictated the use of digital computers for computational and decision making processes in a real time environment. In this respect the advent of the minicomputer offered designers an extensive range of hardware and software capabilities to a wide range of engineering applications. With advancing technology the minicomputer had become more efficient in terms of operation and costs to the point where they gained a relevant popularity in all engineering disciplines.

The evolution of microprocessor technology from an applications point of view came as an extension of the minicomputer to a wider and less costly area of implementation, thus the adaptation to microcomputers from minicomputers is natural. Furthermore the coalition of the two systems, both having similar foundations may be advantageous; such as distributed processing and microprocessor development systems.

Microprocessors are displaying performance capabilities previously matched by TTL and minicomputer designs. The results have been a wave of product realizations with greater cost effectiveness and complexity. Future generations of microprocessors, single component microcomputers, memories and peripherals will ultimately push performance and lower components costs. However the key problem that will remain in configuring microprocessor systems will be the generation of software and in particular its integration with the system hardware. First generation designs were commonly plagued by lengthly development times and insufficient testing capabilities. Albeit a radical transition from traditional TTL designs, new development tools become a neccessity for any microprocessor based application. The effectiveness of producing a viable product will depend on the efficiency of development as well as manufacture and service.

1.2 PRESENT MICROPROCESSOR DEVELOPMENT AIDS

The intent of any microprocessor development aid is for the simplification of a particular segment of the software and hardware design. Various aids are offered by several manufacturers so that microprocessors would be more accessable and usable by a prospective user. Available in several packages these items range in complexity;they include parts families, single board computers, hardware development aides, in circuit emulators and full development systems.

In choosing one of these packages for designing a specific microprocessor application the user need consider the following; the complexity of the design in terms of hardware and software, the amount of hardware and software interaction, the amount of testing required and the flexibility of the package for future use. A brief look at the contemporary spectrum of devices might give some insight to the requirements of a general purpose development system.A detailed description of these devices may be found in (3).

A generalized range of development tools presently offered is depicted in Figure 1.1.Due to the interactive hardware and software nature of microprocessor

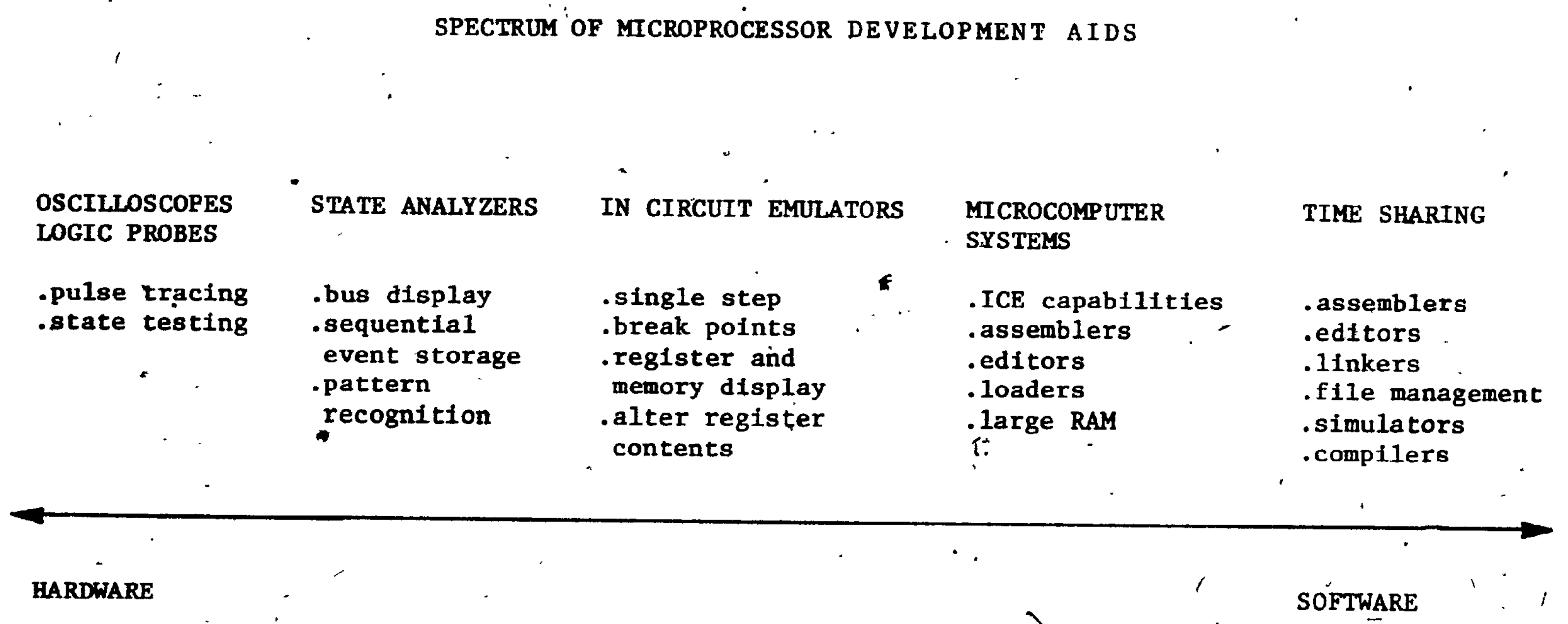


Figure 1.1 A wide range of microprocessor development aids are currently available. Due to the interactive nature of the hardware and software, these aids vary from circuit testing instruments to time sharing computer programs.

systems, it is not surprising that the scope of aids range from time-sharing and minicomputer systems to hardware sequential devices. The extreme left of the spectrum represents hardware support for pulse tracing and static state testing. This equipment such as various oscilloscopes and logic testers which normally would be present in most research labs,are particularly useful for debugging specific logic functions once a problem has been pin-pointed. However for the tracing of software flow in conjuction with monitoring line activity, the use of these instruments is unpractical.

As an extension of the oscilloscope, logic state analyzers allowing multichannel synchronous observation of bus lines, have been aimed for use with microprocessor systems. Multiple display formats (binary, hex,octal,voltage levels) and storage capabilities, allow the user to follow a particular segment of program flow and line activity in real time.These instruments are rather sophisticated and are only useful for testing and debugging logical functions and program segments. Although these devices are adaptable to any microprocessor or logic system they are nevertheless expensive.

Development tools intended for the use for a particular CPU include a range of microprocessor analyzers and In-Circuit-Emulators(ICE). Both systems allow the monitoring of program flow, displaying instructions, addresses, and logical states for handshake lines or other external logic. . Extra features may include observation and alteration of memory and register contents, setting of break points, and system capabilites for more than one make of micrprocessor.A desirable characteristic of these systems is that they provide the user an easy to use push button instrument, allowing the designer to concentrate on the application software and hardware. The main drawback, however is that the devices are unpractical for developing complicated software, and adaptability to a certain microprocessor family is dependant on the "personality modules" offered by the instrument manufacturer.

At the extreme software end of the spectrum are the time-sharing systems run on a host computer. Using cross assemblers, software developmemt for a specific microprocessor is backed by the full software support of the host computer, i,e. editors, assemblers, linkers, high level languages such as FORTRAN, etc. In some cases time-sharing may be advantageous in that it may support several programmers working in parallel. However, the use of time-sharing systems is particularly useful for developing static software-i.e., program variables are time independant. Debugging of real time functions may be achieved through the use of of software simulators, however their performance only seconds real time execution and monitoring. Furthermore the need for simulators is unneccessary if ICE capabilities are existant.

The requirements for a general purpose user orientated development system falls towards the middle of the spectrum which is highlighted by the micro-computer/ICE systems. Their area of application is aimed at hardware and software devlopment and test.

1.3 MICROPROCESSOR DEVELOPMENT SYSTEMS

The microprocessor development system has evolved to be a valuable tool in the design of any microprocessor based application.Microcomputer system designers have witnessed the growth of available systems over the last several years, when Intel introduced the first microcomputer development system for its 8080 family.Indeed development systems now exist for all established microcomputer families supplied both by the major manufacturers of these lines and by independant instrument manufacturers. Naming a few, development systems are available in various formats from Intel, Zilog, Motorola, RCA, TI, Rockwell, Signetics and Tektronix .

There exist two basic modes in which these development systems may be utilized. The first method entails using the development system for software development. Once the software has been been developed and optimized the neccessary hardware design is implemented into the final realization. The second method involves the simultaneous develpment of both software and hardware requirements using the development system for diagnosis and control. For applications characterized by smaller or less sophisticated systems where there is not a considerable amount of interaction between the software and hardware, the first method is adequate. However, for more complicated designs the user is apt to conform to the second method requiring an interactive develpment system. A generalized format for implementing such a system for a specific application is depicted in Figure 1.2. Once the functional requirements of the design have been defined, software and hardware development (including some intermediate testing) may proceed in parallel. Program development incorporates the system's software support which would normally include a resident assembler, editor, and a reasonable amount of RAM with perhaps a floppy disc operating system. An overview may be found in (2). In circuit emulation (ICE) is particularily useful for debugging software and testing software and hardware interactions.

At the present stage of development, various firms offer their own respective family of devices so that no two microprocessors are completely compatible with each other. Although such versatility in architecture and languages offers the user a wide range of

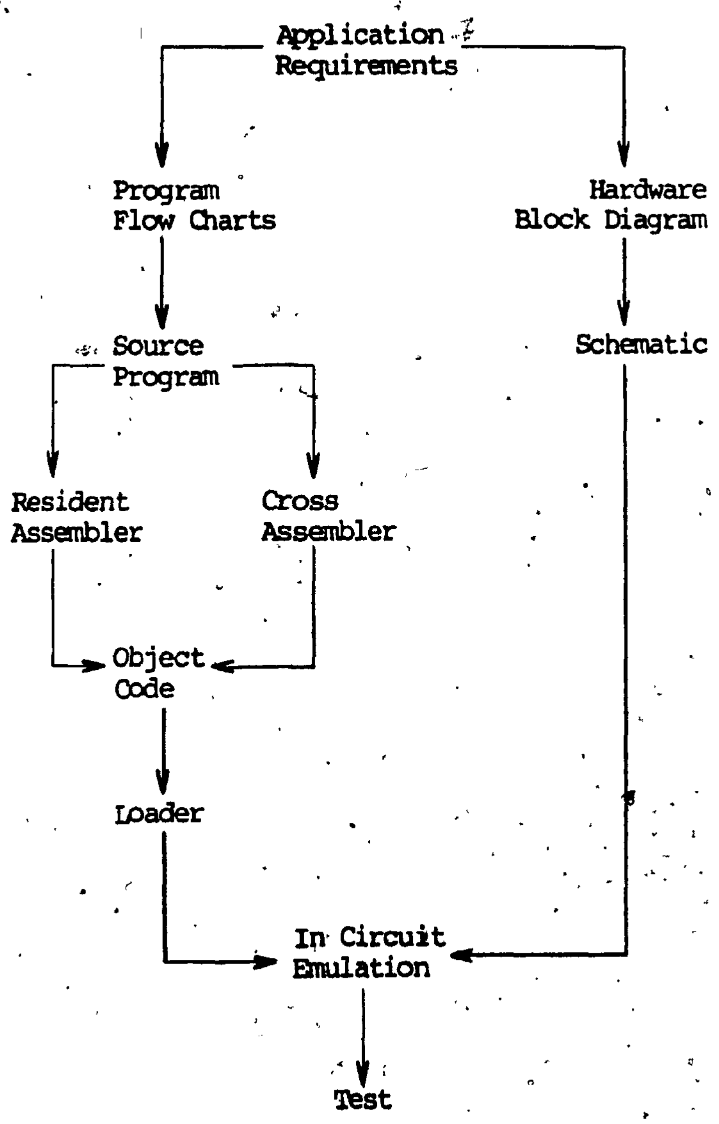


Figure 1.2 Parallel hardware and software development for a specific application. Not indicated is the feedback process for discovering errors at the different stages.

devices to best suit the need, it poses problems to those who wish to configure systems around more than one brand of microprocessor. Thus with the aquirement of a dedicated development system, the designer is committed to a particular make of microprocessor. In terms of financial resources, resorting to numerous development systems for several microprocessors is highly unpractical.

Manufacturers have been aware of this problem and their response signifies the latest trend, which offers optional personality modules for use with various processors. These personality modules usually take the form of hardware interfaces and cross assemblers for adapting the sytem to a number of microprocessor devices.. A good example of such a system is that offered by Tektronix which has the capability of supporting the 8080, 8085, Z-80 and 6800 microprocessors. Although these systems are an improvement over dedicated systems they are nevertheless quite expensive, usually in the $15,000-$20,000 bracket.

CHAPTER 2

AN ALTERNATIVE APPROACH TO THE DESIGN OF A MICROPROCESSOR DEVELOPMENT SYSTEM

2.1 USING MINICOMPUTERS FOR MICROPROCESSORS SYSTEM DEVELOPMENT

If the aids provided for a specific system are unsatisfactory the alternative remains to design ones own personal system. The procurment of a development system around a minicomputer entails the design of hardware interfaces and ~~the~~ writing of the development system software. The initial design time before the system becomes operable, is an important factor to be considered when adopting the building rather than buying approach.The choice will depend to some extent on the availability and experience of the personel required for the project. The minimum requirement would probably include a reasonable amount of knowledge in programming and hardware logical design.

The initial cost outlay for a minicomputer and additional hardware falls within the ball park range of what would be expected for investment into an existing system. Recently minicomputer manufacturers have offered systems for lower end applications at lower costs yet maintain system compatability with the family line. As an example the acquisition of a PDP 11/03 system carries the full software support of the 11 family of minicomputers as well as a host of peripherals. All told a suitable system, as indicated in Table 1 could be purchased for approximately $15,000.

Considerable cost savings could be achieved if an existing minicomputer is already available, as is the case with many groups entering the field of micrprocessor design.The concept of developing microprocessor software on a host computer has existed now for a number of years and is well documented in the literature(1,4,5,6,12). Using the minicomputer software capabilities (e.g. MACRO 11 Assembler) such development should pose no real problem to designer who has some experience in assembler programming. As far as implementing the necessary hardware, microprocessor manufacturers usually provide a generous amount of information on the device operation, interfacing as well as applications. Furthermore, the interfacing of the device into the overall development system provides the designer with experience that is useful

ITEM	PRICE
"naked" Minicomputer	$3,000
Cartridge Disk	$8,000
CRT terminal	$1,000
Line Printer	$1,000
Interfaces	$2,000
TOTAL	$15,000

Table 2.1 Approximate minimal system price list

later on when an actual application is in mind.

In summary, although the approach to such a design involves the extra effort until the development system becomes on-line, the solution for a flexible and efficient system pays off in the long run. The intent of the design goes beyond the notion of simply building and running an operational system but also keeps the following objectives in mind.

<u>FLEXIBILITY</u>-the flexibility of the system configuration makes it adaptable to any current microprocessor device; thus the system is upgradable as technology produces enhanced devices. Functional modules could further be annexed such as PROM programmers and a monitor memory for use in real time in circuit monitoring, as demonstrated in (11). The master-slave concept may be extended for multiprocessor development(i.e. each microprocessor defined by its own hardware and software modules) as well as time sharing capabilities. These ideas are important considerations for implementing the system in an educational environment as proposed in (10).

<u>SOFTWARE SUPPORT</u>-the extent of the minicomputer software support exceeds that of present day development systems. Granted existing systems have been aimed at

a specific application, but on a per dollar basis the minicomputer operating system offers a more flexible and proficient software package.

The operating system is incorporated for the development of system and application programming. Higher level languages may find their use in the construction of handlers, monitors etc. Editors and assemblers are manipulated for application software development. The file management system offers complete storage and referencing capabilities for all files. The extension of the software support allows various functions such as a foreground/background monitor for time-sharing and multiprocessor applications. Naturally the minicomputer could be incorporated in numerous other domains as well, so that the system is not tied down only to microprocessor development.

<u>SPEED</u>-the interactive nature of the minicomputer CPU and disk controller allows for fast assembly of the source application program. During software development, modifications to the microprocessor source program is achieved through editting and reassembly.The inherent speed of the system thus avoids the programmer being subjected to lengthy assembly times and

probable frustrations.

EASE OF OPERATION-the complete sytems serves as a valuable tool for microprocessor applications. The prospective user need not be concerned with the operation of the system but on how to manipulate it for a given application. Eventually the user would require a minimal knowledge of the system editor as well as basic protocol for assembly loading and monitoring. Complete activity may be channelled via standard CRT terminal offering visual push button interaction between the user and system.

2.2 SCOPE OF THESIS

The thesis presented herein concerns itself with one possible implementation of a minicomputer based microprocessor development system. Specifically our discussion will revolve around a PDP 11/45 minicomputer and the RCA 1802 microprocessor. By delving into the global operation and detailed hardware and software aspects of the design,an overall appreciation of the system organization as well as a funtional description of the various elements are supplied.

Presently the laboratory at Concordia incorporates a PDP 11/45 minicomputer with refresh graphics terminial, Decwriter, two RK05 disk drives, and line printer. For a number of years the computing system had been involved in several analytical as well as real time applications. Considering the advantages of using a minicomputer as a microprocessor development tool, the incorporation of the 11/45 was for us at least the most direct and economical approach. However, since the 11 family is software compatible down the line,the system is adaptable to any member of the PDP 11 family.

The generalized system configuration is illustrated in Figure 2.1. Transfer and control of parallel data to and from the PDP is handled by the DR-11C general purpose interface. The module which is purchased from DEC for approximately $800 can easily be configured for a wide range of I/O use. Via the Controller the input and output buffer registers of the DR-11C (DRIN, DROUT) are channelled onto a single 16 bit bidirectional bus. Bus control may be requisitioned to any of 1 of 8 peripherals I/O cards and retains control until deselected. As an extension of the DR11C, selection and transfer of data to the peripheral I/O cards is under complete software control.

In the case of the microprocessor development system one of the I/O cards will be represented by a particular microprocessor Monitor module. Its primary function is for the control, loading and monitoring of the microprocessor. Associated with each microprocessor Monitor card will be its appropriate software module.Thus several micro processors whether identical or not will run with their own Monitor interfaces, software modules and card addresses on the Controller bus either singularily or simultaneously

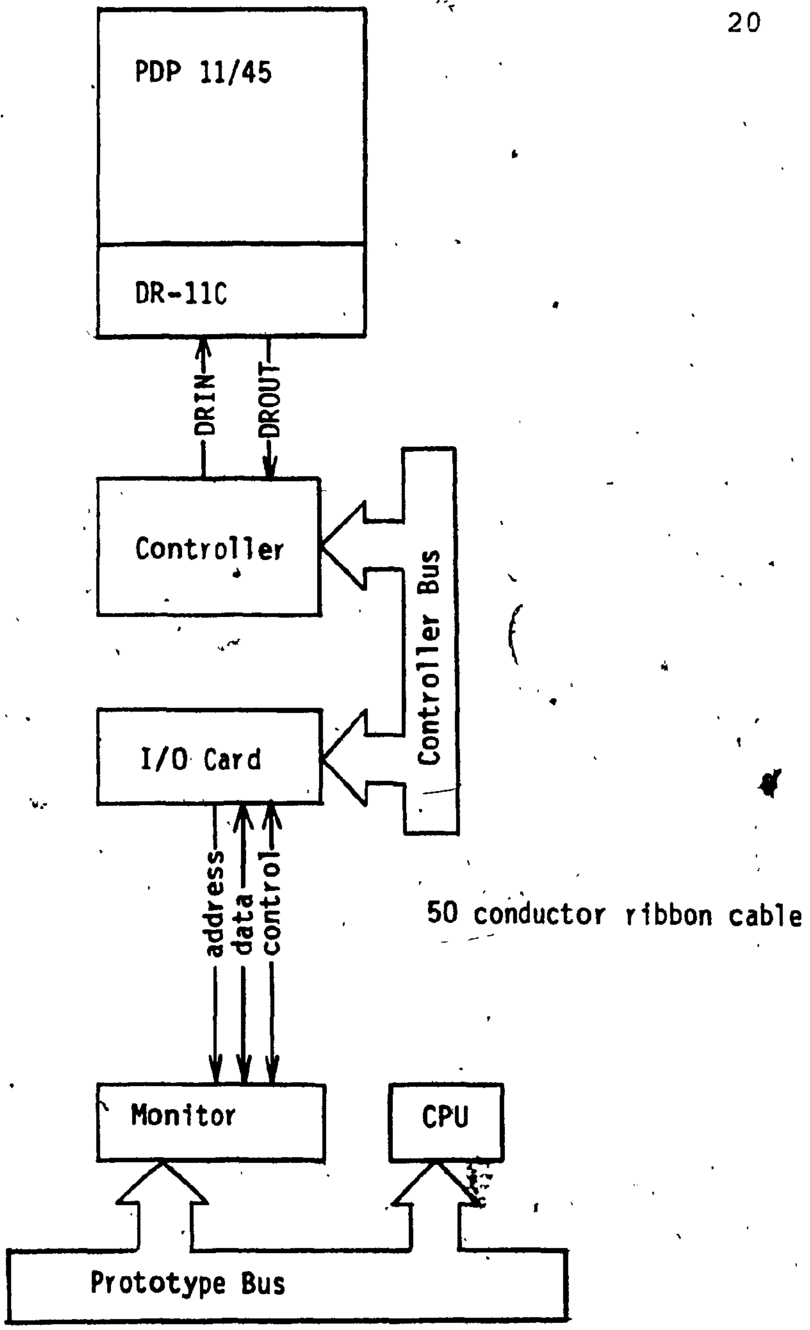


Figure 2.1 System configuration. By linking the mini-computer and prototype, application programs are loaded, monitored and run.

multi-user environment).

The development system is built around a modular basis, i.e. CPU and Monitor printed circuit cards are designed for prototype integration. Additional plug in modules for various support functions and I/O may be developed and retained for future applications. The same principle holds for commonly used software routines and macros which are linked into the final application routine. During development the monitor module is part of the prototype system and is useful for loading programs, in-circuit-emulation and debugging.Once the hardware and software have been developed and optimized the monitor is removed and is retained for future applications or maintenance checks on the product.

CHAPTER 3

THE HARDWARE

3.1 INTRODUCTION

The hardware requirements for the development system may be subdivided into three basic groups; these include 1) I/O from the master, 2) target machine , and 3) link with the prototype system.

I/O from the masters defines how status,control and data are transmitted from the master minicomputer. The target machine defines the structure of the microprocessor or microcomputer under development. Finally linkage specifies how information is passed between the master I/O and the target machine.

In our application the I/O from the master is handled by the Controller and the specialized DR-11C interface to the minicomputer. Our target machine is represented by the RCA CDP 1802 microprocessor, integrated into the CPU module. Linkage is accomplished via the Monitor module. A complete description of the various hardware elements are given in the following sections.

3.2 THE DR-11C

The DR-11C (Figure 3.1) is a general purpose interface between the PDP-11 bus and a user's device. The interface provides buffer/registers for input and output program controlled parallel data transfer, as well as for status and control functions. The status lines may be controlled by either the program or external device for command, monitoring and interrupt functions. The interface consists of three registers; control and status (DRCSR), input buffer (DRIN) and output buffer/ register (DROUT). Operation is initialized under program control by addressing the DR-11C to specify the register and type of operation to be performed .In order to synchronize timing of data transfers to and from the interface, two control signals New Data Ready (NDR) and Data Transmitted (DTR) are available to the user.

Upon software transfer of data to the DROUT the NDR pulse is strobed indicating the user's device that data has been loaded. When an input from the DRIN has been executed the DTR line is strobed indicating that data has been read by the 11/45 CPU. The six bits of

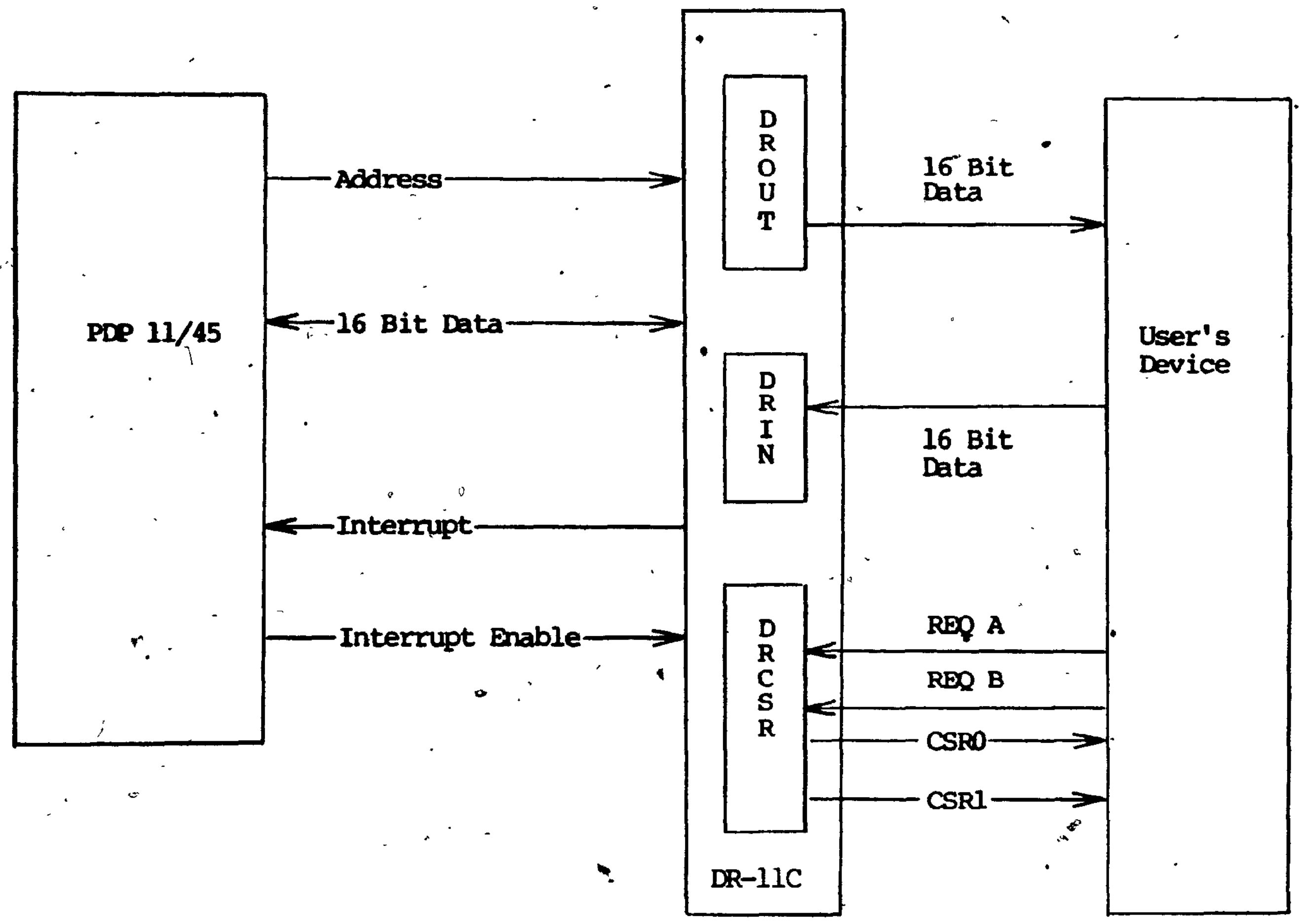


Figure 3.1 . The DR-11C is a general purpose interface between the PDP 11/45 minicomputer UNIBUS and a user's peripheral.

the DRCSR are used for control and monitoring functions of the device. The two request lines REQ A and REQ B, which can only be read by the program, are manipulated either to initiate interrupts (depending whether INT A and/or INT B are set)or to provide flags that may be monitored by the program. The bits CSR0 and CSR1 are set under program control to provide commands to the external device. A complete description of the DR-11C may be found in (8).

3.3 THE CONTROLLER

The transformation from a separate I/O structure of the DR-11C to a more compatible bidirectional bus structure is achieved by the Controller. Housed in the minicomputer mainframe,the Controller channels the I/O and status registers of the DR-11C (DRIN,DROUT,DRCSR) onto the bidirectional bus of the Controller chassis. The detailed schematic of the Controller is given in the appendix.

Bidirectional traffic control as well as bus driving capabilities are handled by the 8216 drivers. Normally the direction of the drivers are in the output mode with respect to the DR-11C , i.e.the DIEN input is low and the Controller bus is hooked to the DROUT. Upon a program Data In sequence the DTR line is strobed. On the rising edge of the strobe two events occur; 1) the DRIN buffers are enabled, 2) direction of the 8216 drivers is switched so that the Controller bus is channelled into the DRIN buffers. At the same time a stretched and inverted version of DATI is issued to the Controller bus.

The Controller has the option of supporting up to

8 I/O peripherals. Selection of a peripheral or I/O Card requires the appropiate decoding of a three bit address. The format of the address word is shown in Figure 3.2.

The address byte is organized around 8 bits of which bits 3-5 are decoded for I/O Card selection. The remaining 5 bits may be used for control functions or for selecting one of 32 ports on the selected I/O card. Address bytes are transmitted on the Controlller bus to each card for decoding. Only one of the cards will match the 3 bit card address and consequently gain control of the bus or perform some function depending on the setting of the remaining address bits. Data (up to 16 bits) and address (8bits) words are multiplexed on the Controller bus in accordance with the appropiate pulsing of the DATO and ADDR strobe lines. To differentiate between an address or data word being transmitted the CSR0 and CSR1 bits of the DRCSR are used. Thus during a Data Out program sequence, depending on how these two bits have been set will determine whether to issue a DATO or ADDR strobe. This is accomplished by decoding the two CSR bits

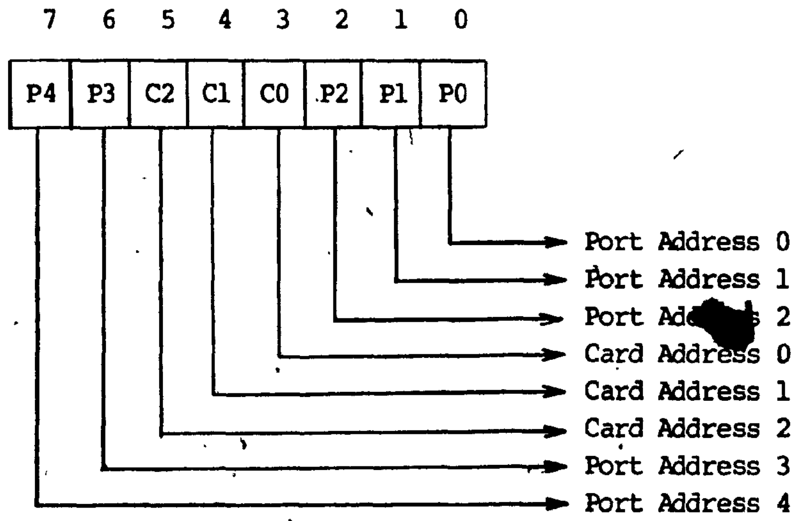


Figure 3.2 Controller address byte format

with the NDR signal by means of the 7242 decoder. The remaining bit combinations of the CSR bits allow for two more functions. The first, INIT, will generate a strobe which may be used for the initialization or resetting of a device or operation.The second, DATA HOLD, will reverse the normal direction of the Controller bus drivers i.e. place the 8216 drivers on the input mode with respect to the DR-11C. This particular function is useful for observing the Controller's bus lines under static conditions. A summary of the CSR functions are given in Table 3.1.

Lastly the Controller buffers and inverts the REQ A and REQ B lines before feeding them into the DRCSR. Linkage between the Controller chassis and the DR-11C Berg connector is accomplished via a 40 connector ribbon cable . The Controller has the capabilities of supporting 8 (8"X 10",40 pin) plug-in modules, and maintains its own 5V, 15V@1A power supply.

CSR0	CSR1	Function
0	0	DROUT contains address
0	1	DROUT contains data
1	0	INIT strobe
1	1	IN mode

Table 3.1 CSR0 CSR1 truth table.

3.4 THE RCA CDP 1802 MICROPROCESSOR

The first step in any microcomputer system design is to develop a set of criteria for microprocessor selection. Some of these criteria include;

-flexibility

-performance

-cost

-peripheral compatability

-reliability

While the development system is adaptable to any device, our intention was to carry the basic philosophy of flexibility one step further by choosing a microprocessor that would be compatible to a wide spectrum of applications. In considering various devices the RCA CDP1802 8 bit CMOS microprocessor was chosen as central control and processing unit.

The range of microprocessor families and their immediate application domains is shown in Figure 3.3. In contrast to the 4 bit and 16 bit devices, system architecture around 8 bit machines appear to cover the most ground. Nevertheless, enhanced 8 bit devices can

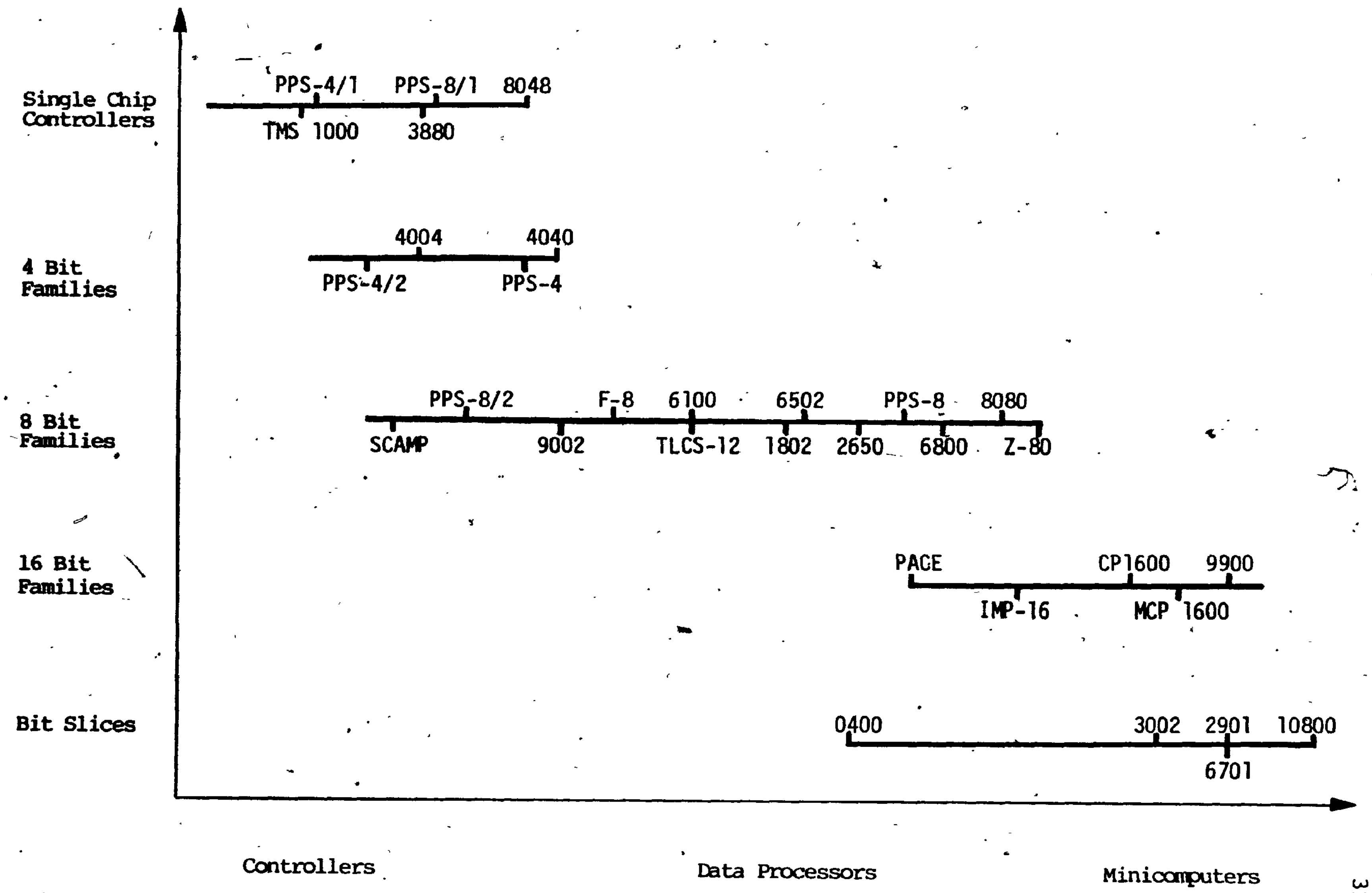


Figure 3.3 Range of processor chips and their application domains.

handle 16 bit data, making it possible for the user to upgrade his system .

In the applications spectrum the Intel 8080 and Motorola 6800 are very popular processors. Both are similarily organized with bus configurations typical of 8 bit systems. Such bus organization allows interface and peripheral IC's to use these lines, ensuring the user a well formulated method of designing his basic system.

Our decision was to proceed with the 1802 microprocessor manufactured by the RCA Solid State Division. The device possesses many of the advantages of the older established families as well as several of its own useful characteristics. In brief RCA 1802 features include:

-single chip CPU

-single 5V supply required

-65K memory access

-CMOS technology

-2.5 microsecond instruction cycle

-16 scratchpad registers(may point to either address or data)

-91 instruction set

-on board DMA facility

-interrupt and programmed IO capabilities.

The block diagram of the 1802 architecture is shown in Figure 3.4. Furthermore the device will be pin to pin compatible to an enhanced SOS (Silicon On Saphire) version to be released by RCA at some later date. The SOS device promises to push speed through-put by a factor of two. Complete performance specifications and functional description of the 1802 microprocessor is given in (9).

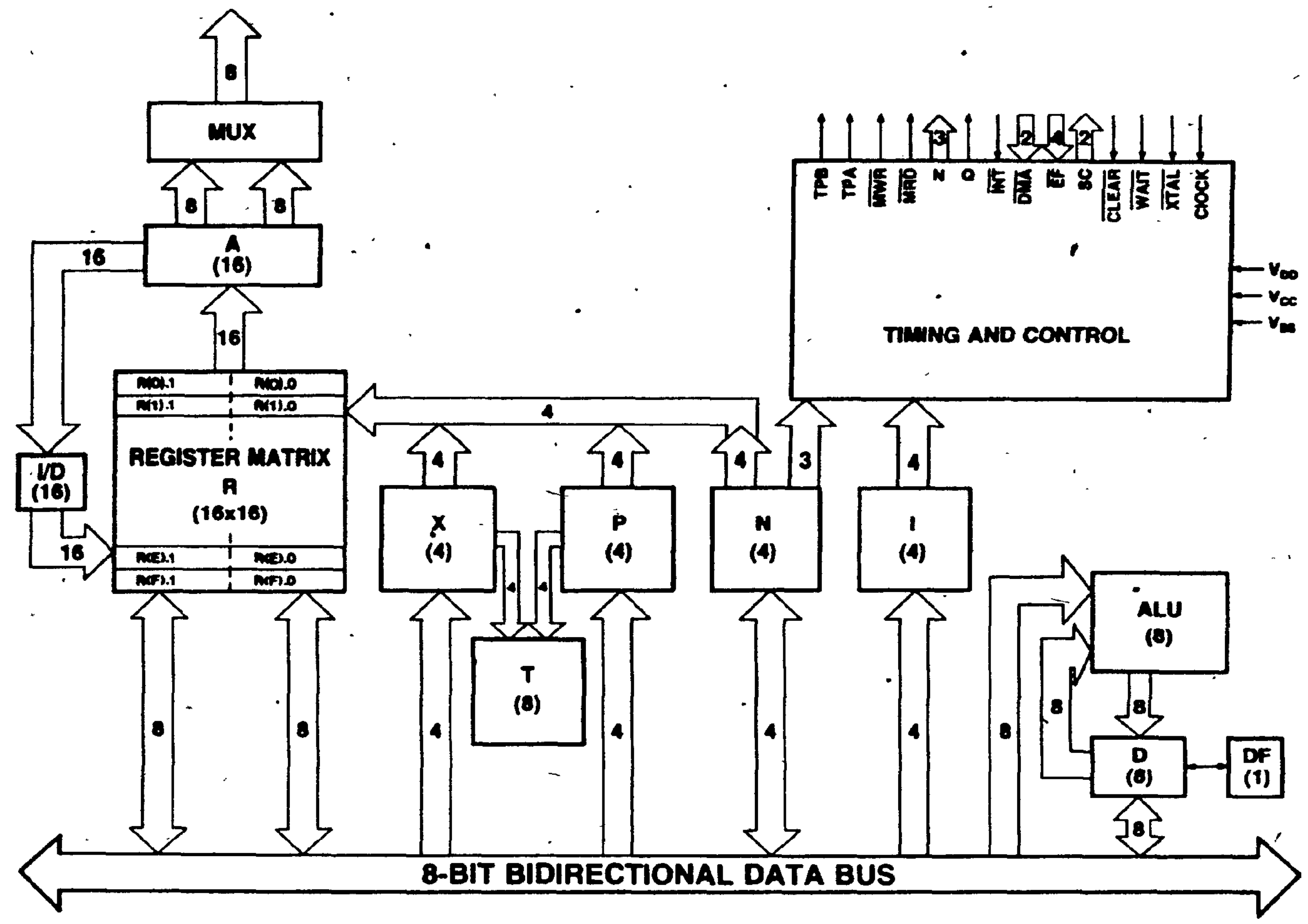


Figure 3.4 RCA 1802 microprocessor architecture.

3.5 CPU MODULE

The 1802 microprocessor is incorporated into the CPU module. The module also contains 1Kbytes of on board memory which may be configured as either RAM and/or PROM. The CPU module block diagram is illustrated in Figure 3.5. For initial development,RAM may be used and upon final prototype development the required software may be permanently loaded into PROM. The module is packaged on a standard size PC board and comprises the fundamental unit for the application development and final product.The schematic for the CPU module is given in the appendix. Maximum component density was considered in the initial design. The result is a high density single board microcomputer. Although the 1802 contains an on chip clock oscillator (requiring and external crystal), an external clock may also be provided.The latter course was taken in the design (using the 74123) the intent being that the user may set and adjust the operating clock frequency (via an adjustable trim-pot)for optimum performance. 1802 specs allow for a maximum 6.4 MHz clock frequency (operating at a 10V supply). All data and control

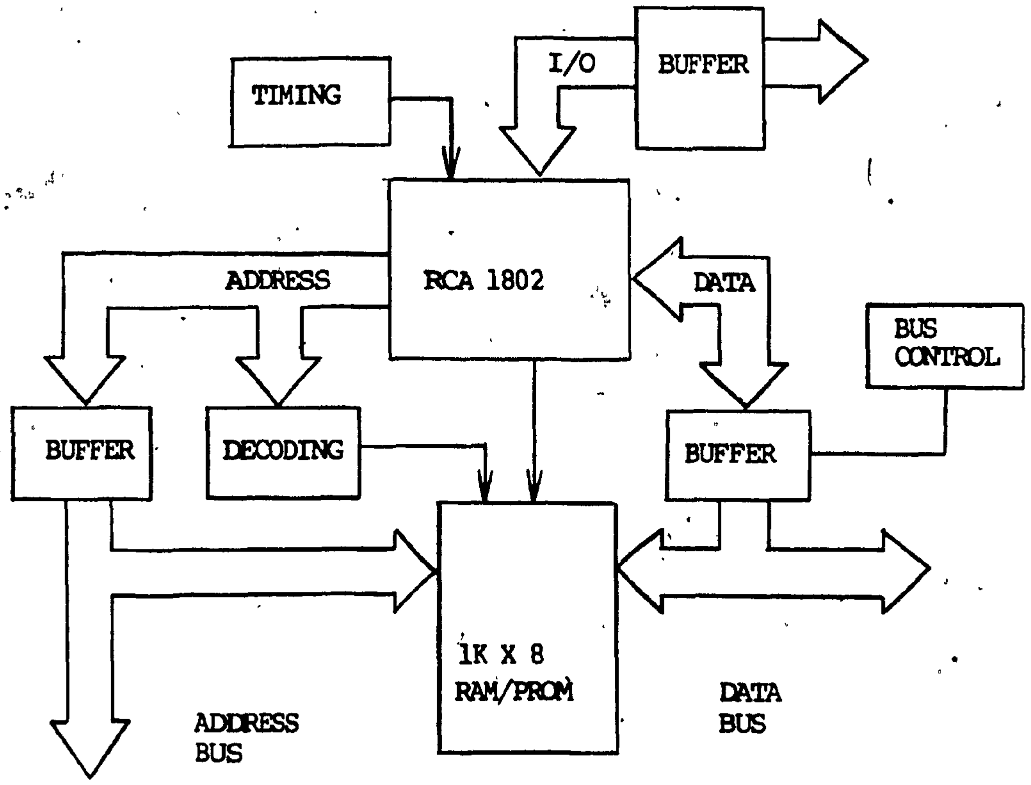


Figure 3.5 CPU module block diagram. Based on the 1802 microprocessor and 1K memory (RAM and/or PROM) the module serves as common denominator for all applications.

outputs are buffered . Control inputs (flags, interrupts, DMA requests, mode control) are pulled up to 5V through pull up resistor networks.

On board memory is configured around 256 X 4bit static devices.Eight components are incorporated to yeild 1Kbytes of memory. The Intel 2112 RAM and 3641 PROM are used. Pin compatability of two devices allows for interchangability and offers various combinations of RAM and /or PROM configurations.

The 1802 provides 8 memory address lines, supplying a 16 bit address word in the form of two successive address bytes. The high order byte (A1)appears on the address lines first followed by the low order address byte (A2). Two timing signals TBA and TBB are supplied by the microprocessor each machine cycle for synchronizing external events to internal 1802 events. The relationship between TPA,TPB and the memory address lines are shown in Figure 3.9.

The CMOS 4042 latch is used to record the most significant as well as lower 3 bits of the high address byte (clocking occurring on the falling edge of TPA). Decoding of the lower three bits of A1 is accomplished by the 8205 selection switch to choose the

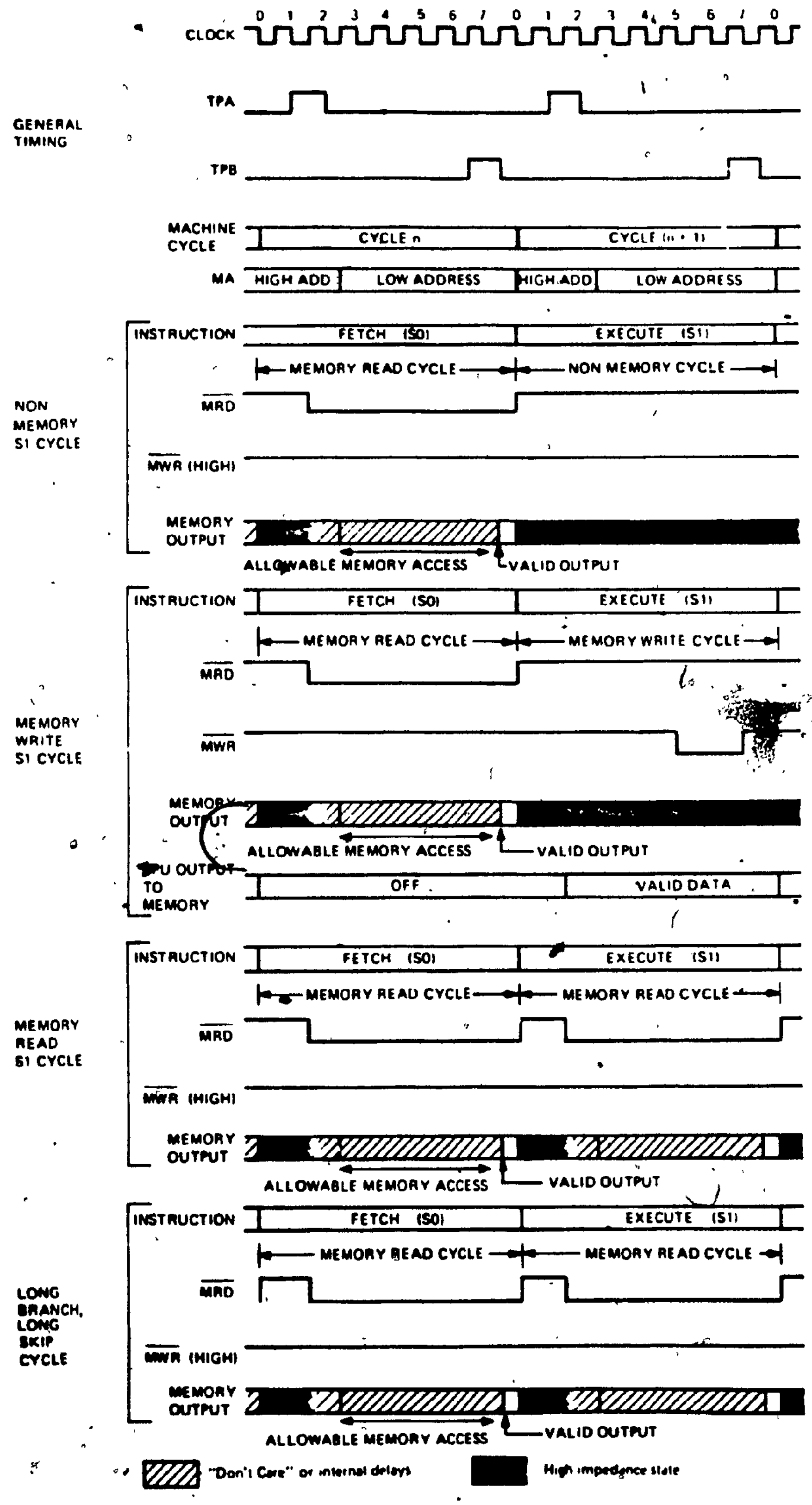


Figure 3.6 1802 timing waveforms.

appropriate memory page(1 page equals 256 bytes). Although up to 2Kbyte of memory may be accessed only 1Kbyte of on board memory is actually available. Bit 7 of the high address byte, if set deactivates the 8205 decoder and drives the IOEN line active low.

The memory data output (see Figue 3.9) is latched into the CPU on the falling edge of TPB . For a 3.2 MHz clock frequency (max at 5V) the allowable memory access time is in the order of 1.4us which is suitable for the maximum access time of lus of the 2112 static RAM device (the 3621 PROM will have a faster access time of approximately 60 ns).

Setting of S1 switch accordingly for each page determines whether RAM or PROM configuration is being used. For the PROM configuration S1 for the specific page is set in the OPEN condition and chip selects the device. Page selection via chip select is derived from the 8205 decoder. For the RAM configuration S1 for the specific page is set in the CLOSED position. Consequently the MWR line is fed to the R/W line of of the 2112; when low indicates a Memory Write operation. Note from Figure 3.6 that data from the CPU is stable on the bus prior to activation of the MWR line. Normally the 2112 is in the READ state i.e. the R/W line is maintained high. Once again page selection is accomplished through the chip select derived from the 8205 decoder. Data may be loaded from the CPU to an external device in the same manner as it is loaded into memory. An I/O device may be characterized by a specific address or address block and programmed data transfers to these loactions will transmit data to the device. Presently, the most significant address bit (bit 7 of A1) is used to determine whether an address is referencing memory or an I/O device. The MSB of the 16 bit address is latched at TPA time which in turn enables the IOEN line (active low) and deselects the on board 1K of memory.

The 64K (65,535) memory referencing capability of the 1802 is sectored in the following memory mapping scheme . Full memory is divided via bit 16 into 32K for memory and 32K for I/O.Should the full memory address space be required the IOEN option may be disabled through jumper J1.

Memory is organized around pages 256 bytes long. Eight pages are integrated into a memory bank (2K). Memory extension cards are organized around memory banks, and possess the appropiate decoding for page selection. Recall the address bits 8-10 of the address word are used to select the proper page. The remaining address bits 11-14 will choose 1 out of the 16 possible memory banks.

The on board 1K memory on the CPU card represents one half the capacity of a possible memory bank. The full memory bank may be completed by stationing the remainig 1K elsewhere. An added input line to the module is supplied, MEMDIS which when activated deselects the on board memory. Presently the 1802 will address the first 1K of memory directly to the on board memory; however through the use of the MEMDIS line this 1K may be relocated elsewhere. In situations where external memory (banks) are being utilized an external decoding circuitry will define the relative locations of the various memory banks. Thus each memory bank module may possess a similar MEMDIS input and depending on the state of the line decides if the given bank is selected.

Under normal operation the 1802 may be in one of any four machine states. Each machine state designated S0-S4, lasts the duration of 8 clock cycles, comprising one machine cycle. Two state code lines SC0, SC1 are available to the user and indicates that the CPU is ; 1)fetching an instruction, or 2)executing an instruction, or 3)processing a DMA request, or 4) ackowledging an interrupt request. The State Code truth table is shown in the 1802 signal description chart in Figure 3.7.

SC0,SC1,Q and N lines are buffered via the 7432 gates. The Q line is a software generated output signal and the N lines are activated by COSMAC I/O instructions. The 8 data lines of the 1802 are configured in the bidirectional mode. Data output is channelled through the 8216 bidirectional bus drivers. Although outputs from the 1802 device can interface directly to a single TTL load, the 50ma driving capabilities of the 8216 drivers are well suited for bus driving. The direction of the data traffic is dependant on the type of instruction being executed. In general when the MRD line is active data is being read from memory or an I/O device. Under such conditions the 8216 transceivers are in the input mode with respect with the CPU. An exeption to the rule is the INPUT instruction which loads memory and an internal

CPU registers simultaneously from the data bus. In such a situation MRD is high, however the 8216 must be in the input mode (i.e. the bus lines must be connected to the 1802 data bus). Extra logic has been provided to switch bus directions for the IN N instructions.

During DMA (S2) cycles data is channelled directly to and from memory via the data bus. The source or destination of the data is from to or from some external peripheral, however the 1802 yeilds the corresponding memory addresses and monitors the transactions. Once a DMA cycle has been ascertained the microprocessor data lines are disabled from the card data bus via the CS line on the 8216 bus drivers. State decoding logic is activated during an S2 cycle to disable the D4 and C3 drivers. Furthermore S2 cycles are indicated by the S2 active output line.

In order to inform any external device the present direction of data flow on the CPU data bus, The DATIN ouput is provided. Direction of the bus will be instruction and/or state dependant.

Nine input lines allow for I/O controllers to transfer status or commands to the central processor. All inputs are pulled up to 5V via resistor networks.

Active low inputs to these lines may be derived from one or more devices through the use of open collector drivers.

Four input flags enable a device to transfer status information to the microprocessor. The flags are internally sampled at the begining of each S1 cycle and may be tested by conditional branch instructions.

Activation of the INTE interrupt line causes the machine to enter the S3 state. During this cycle the present program counter is stored and replaced with the starting address of the interrupt routine. Interrupt requests may be enabled or disabled by an internal flip-flop set or reset through the software. As soon as the microprocesor enters an the interrupt state the interrupt enable flip-flop is enabled to inhibit any further interrupts.

Direct memory transfer signals DMAIN and DMAOUT are user generated signals that may be ascerted at any time. Data is transferred directly via memory data bus, and address locations are stored and incremented automatically by an internal register (R0). The DMA as well as the INTE lines may request service independant of the microprocessor internal events, however the CPU will first complete its present S1 execution cycle before going into either an S2 or S3 state.

The remaining two inputs CLEAR and WAIT are decoded by the 1802 to place the CPU in one of any four operational modes. The four modes of operation; LOAD, RESET, PAUSE, and RUN and corresponding CLEAR and WAIT truth table are listed in Figure 3.7

In the LOAD mode a program may be loaded directly into memory. The process is similar to a DMAIN operation with the exception that the intervals between transfers are filled with IDLE (no-op) operations.

RESET resets the internal registers, TPA and TPB are suppressed and the the CPU is placed in the S1 state.

PAUSE stops the internal CPU timing generator on the falling edge of the clock (of the upcomming full clock period). Although the clock may be left running, the CPU is frozen in the state at which time the PAUSE occured. Resumption of operation occurs when the PAUSE condition is removed.

RUN places the machine in its running state. When initiated from RESET, RUN will begin program execution

Signal Descriptions

BUS_{0-7}
8-bit bidirectional data bus for CPU - memory - I/O transfers.

N_{0-2}
I/O selection lines. The 3 lowest-order bits of the N register appear on these lines when an input or output instruction is executed They are used to uniquely select the desired I/O port. Different I/O instructions generate different signals on the N- lines (op codes 61-67 are outputs, 69-6F are inputs).

Q
Bit-serial output line that can be set or reset under program control via two dedicated instructions (7B, 7A).

INTERRUPT
Input to the CPU that initiates execution of the user's interrupt service program.

$\overline{\text{DMA IN}}$
$\overline{\text{DMA OUT}}$
Direct Memory Access request lines. Initiates DMA data transfer action within the SCP1802

$\overline{EF_{1-4}}$
Four External Flag lines Conditionally tested by branch instructions to convey external information to the CPU, priortize multiple interrupts, etc.

SC_0, SC_1
The State Code lines output the SCP1802's current machine state.

STATE		SC_0	SC_1
S0	Instruction Fetch	0	0
S1	Instruction Execute	0	1
S2	DMA Transfer	1	0
S3	Interrupt	1	1

$\overline{\text{MRD}}$
Memory Read signal. Can be used to control 3-state outputs from addressed memory if it has a common data in/out bus. Also used to indicate direction of data transfer during an I/O instruction.

CLOCK
Input for external one phase clock signal.

$\overline{\text{XTAL}}$
A crystal is connected between $\overline{\text{XTAL}}$ and CLOCK (typically with a 10 megohm resistor) when the on-chip oscillator is used.

$\overline{\text{WAIT}}$
$\overline{\text{CLEAR}}$
Control lines to determine the SCP1802's mode:

MODE	$\overline{\text{CLEAR}}$	$\overline{\text{WAIT}}$
Run	1	1
Pause	1	0
Reset	0	1
Load	0	0

Run:	Normal CPU instruction execution.
Pause:	Stops CPU operation at the next high-to-low clock input; only the oscillator continues to operate.
Reset:	I, N, Q, X, P, R (0), BUS → 0 IE → 1 (interrupts enabled)
Load:	Allows an I/O device to load memory from location 0000 without a "bootstrap loader".

V_{DD}, V_{CC}, V_{SS}
The I/O voltage supply level V_{CC} is isolated from the internal CPU voltage level V_{DD} This allows for simple interfacing bewteen the SCL1802 and TTL, NMOS, etc. V_{SS} is ground.

TPA
Timing Pulse for strobing high order memory address byte into external latch or ROM.

TPB
Timing Pulse used by external I/O devices to time interaction with CPU.

MA_{0-7}
8-bit Memory Address bus carries the multiplexed 16-bit memory address, high-order byte first followed by the low-order byte.

$\overline{\text{MWR}}$
Memory Write. A pulse appearing after the memory address lines have stabilized.

Figure 3.7 1802 signal descriptions.

starting at location 0.

The CPU module is integrated on a 4.5" X 8" double sided printed circuit board. Maximum density is achieved through the use LSI components as well as resistor networks. Power and ground busses are well distributed and decoupled so as to maintain a relatively noise free environment. CMOS components have been used extensively so as to reduce power requirements and increase noise immunity.

Operating at 5V the module consumes in the order of 80ma.The module may be produced in small quantities for under $100.00.

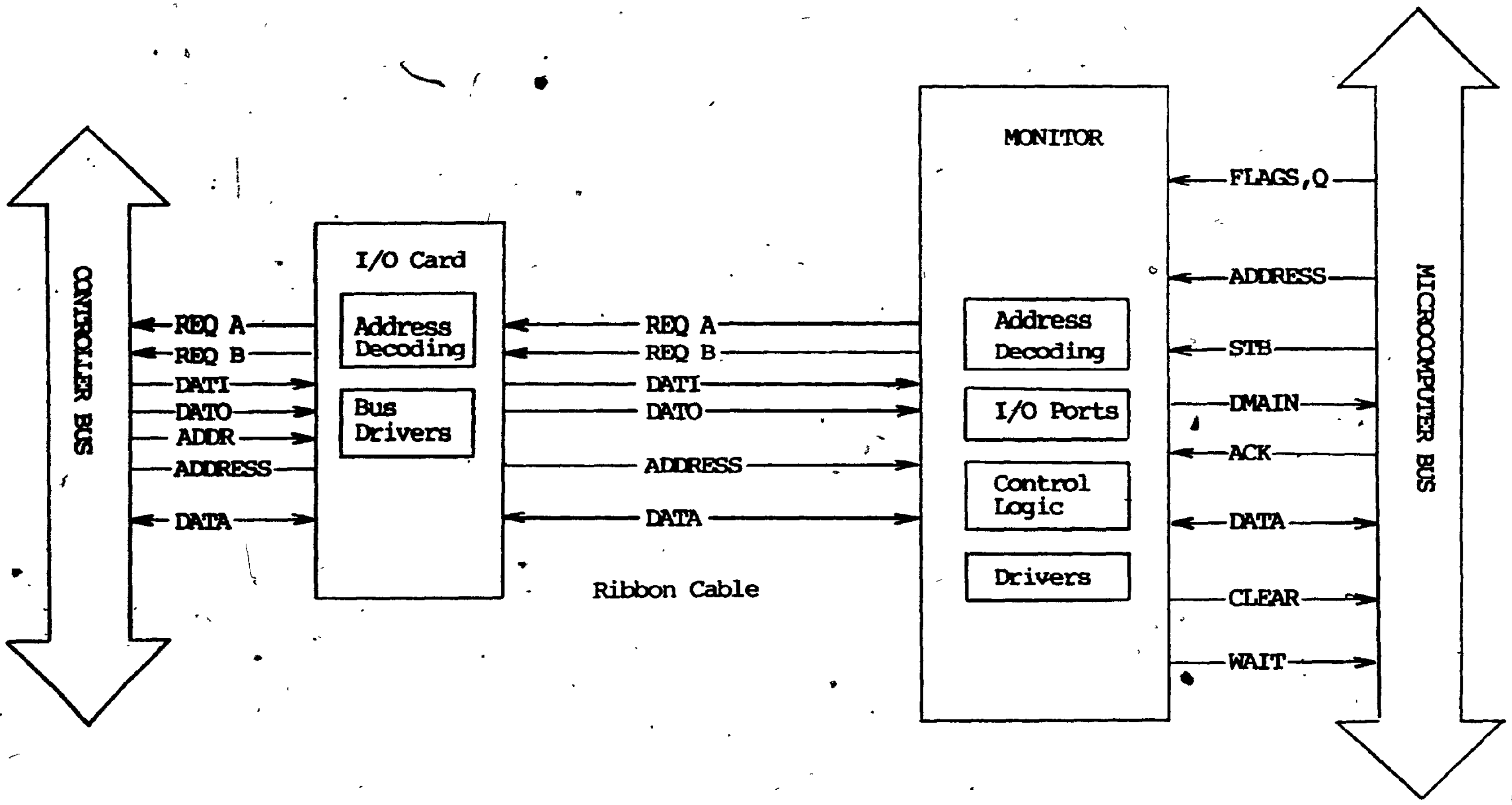
3.6 MONITOR MODULE

The CPU module is interfaced to the I/O Controller via the Monitor module. The Monitor is responsible for the transfer of data and commands to the microprocessor as well as for monitoring the status, address and control lines of the microprocessor.The module is linked by a 40 line ribbon cable ,to its associated card on the I/O Controller chasis.The generalized configuration is illustrated in Figure 3.8.

The I/O card is a simplified version of the address selection logic . Basically it contains a comparator for address matching as well as line drivers for the cable to the Monitor module. The schematic for the monitor is given in the appendix.

Upon card selection the 8216 3-state drivers are enabled thereby hooking the monitor onto the Controller bus, the remaining bits of the address word are latched and finally the individual ports on the Monitor are enabled. Note the 8216's are in the input mode by default and change direction only on the issuance of the DATI strobe.

The Intel 8255 PPI(programmable peripheral int-

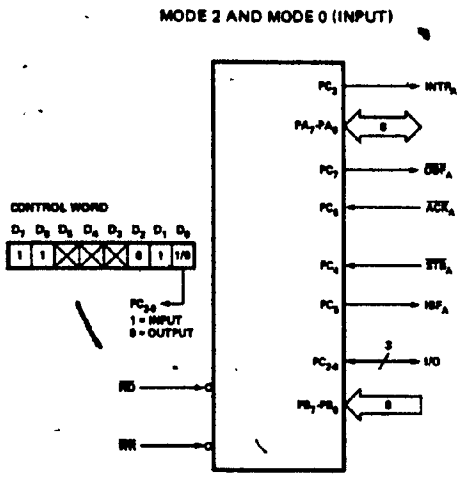


<u>Figure 3.8</u> Software controlled transfer of data and control between the microcomputer and Controller is achieved via the Monitor module.

erface) is used extensively for tramsfer of I/O. The 40 pin package may be configured in several ways depending on the application in mind.

Bits 7 and 6 of the address byte are used for selecting one of the three ports of the 8255 via the A1 and A0 inputs. The lower three bits of the address byte are decoded through the 4028 to choose one of five functions. Four are used in conjunction with the DATI strobe to read one of four input ports.The fifth is incorporated for single stepping the microprocessor through consecutive machine cycles.

Essentially the 8255 multiplexes the Monitor bidirectional data bus to one of three ports (A,B,C). Selection and configuration of the ports is done via a software command word. For our application the PPI is configured in Mode 2 and Mode 0 (Figure 3.9). Port A is used as a bidirectional port for data transfer to and from the the CPU module data bus.PC4-PC7 lines from Port C are used for handshaking of data transmission.Port B is used as a input read only port, monitoring the status of the microprocessor DMA,INTE, N, and state code (SC1,SC0) lines. The remaining three output Port C lines (PC0-PC2) are used in conjunction



<u>Figure 3.9</u> 8255 mode configuration. Port A (mode 2) is a bidirectional 8 bit bus using five lines from Port C for handshaking. Port B (mode 0) is set as an input 8 bit port.

with the microprocessor WAIT and CLEAR lines. Writing data from the Controller bus into the PPI occurs during the DATO strobe. All incoming control signals are buffered .

The destination of the incoming data is determined by the port selection bits A1 and A0 .An incoming byte may either be data (which will be directed to one of the ports) or a control word. The control word is used for initializing the 8255 into its configuration and for setting or resetting 8255 internal flip-flops.

Selection of the 8255 itself as well as any of its internal ports is accomplished using the address byte format (Figure 3.2). P0-P2 set to 0 will select the 8255 via the 4028 decoder. C2-C0 is the actual card address and is strap selectable on the various cards. For our purposes the Monitor module has beeen assigned the address 1.

The microprocessor operational mode, set by the WAIT and CLEAR lines are set by writting the appropriate bit code to PC0 and PC1 of Port C. PC1 is wired through a buffer to the CLEAR line. The WAIT line is tapped from two sources; one is PC0 on the 8255 and secondly the single stepping circuitry via a NAND gate. Normally single stepping is disabled and the 1802 operational mode is derived from PC0 and and PC1 of Port C .

Single stepping is accomplished by placing the CPU in the RUN mode for a single machine cycle and then halting it (placed in the PAUSE mode) until another single step is requested. Refering to Figure 3.7 we note that the RUN and PAUSE states may alternately be achieved by flipping the WAIT line between high and low states.

Initially the CPU is placed in the PAUSE mode. A single step is initiated by using the address byte command. P0-P2 of the address byte are decoded forcing the associated output pin of the 4028 decoder to a high level. The low to high transition is transformed to a positive going pulse which presets the D flip flop. Consequently the Q output falls low pulling the WAIT line high and the CPU is presently in the RUN mode. As soon as TPB appears (ocurring at the end of each cycle) its rising edge forces Q back to a high level.Halting of the CPU will occur on the first falling edge of the full clock pulse which follows the falling of the WAIT line. With respect to TPB, halting of the CPU falls directly on the end of the current machine cycle. Subsequent single stepping will occur when the functional address byte is retransmitted. Repeated transmission of the single stepping address byte will cause incremental single stepping. Note that before and after a single step has been executed the machine is in the PAUSE mode for an indefinite amount of time. To exit the machine from the single stepping mode of operation the correspnding contents of Port C (PC0, PC1) are altered appropriately.

Continuous single stepping will allow program execution, however at a slower rate as would be if the machine were operating in the RUN mode. This pseudorun state of the machine will ultimately depend on the rate at which the single step command is supplied to the Monitor module. This in turn is dependant on the the software module issuing the commands. The objective of the single step mode is that it serves as an important debugging tool. After each increment of the program counter the user may observe the status of the address, data, and control lines. Consequently program flow may be followed step by step and be compared with its functional flowchart. Anomalies may be pin pointed to a certain instruction sequence or condition on the status and control lines. The monitor accesses all data, address and control lines of the microprocessor. These are subsequently relayed to the user by the master software monitor module. Information is tapped from the Monitor's addressable ports. Reading ports A and B of the 8255 would require the appropriate address byte followed by a Data In software sequence (DATI strobe generated by the Controller).

Bidirectional data in Port A may be latched in both directions. Data on the CPU module data bus is latched into Port A on each rising edge of TPB. Strobing of data may be disable by the S2(DMA)condition via the Nand gate .

During each single step new data from the CPU module which is in fact the microprocessor data bus is latched on the rising edge of TPB.During the PAUSE interval,Port A may be read by the software monitor program running on the PDP 11/45. The actual data latched into Port A will be dependant on the machine cycle being executed. Note, however that during a fetch(S0) or DMA (S2) cycle data may either be output from the CPU memory, external device or the data bus may be in the high impedance state.

During a fetch cycle, the memory loaction containing the next instructiuon is accessed. High and low address bytes are strobed into the two 4508 8 bit latches on the falling edge of TPA and TPB respectively. Whether the internal 16 bit address accumulator of the 1802 is the actual program counter or any of the other 15 remaining general pupose registers during a S1 or S2 or S3 cycle,low and high address bytes are latched into the 4508. As an example, if the instruction INC R9 (increment register 9) was being executed, the contents of register 9 would appear in the address accumulator and consequently be latched. High and low address bytes are read by a Data In sequence using their respective address bytes. The complete list of possible address bytes for the Monitor module is given in Table 3.2.

The present state of the EF1-EF4 flag lines and Q line are read through the the 14503 hex buffer. Normally the output of the buffers are in the high impedance state until enabled by the appropriate address byte. No latching is neccessary for these outputs

ADDRESS								FUNCTION
A0	A1	C2	C1	C0	P2	P1	P0	
0	0	0	0	1	0	0	0	Write PORT A
1	0	0	0	1	0	0	0	Write PORT C
1	1	0	0	1	0	0	0	Write CONTROL
0	0	0	0	1	0	0	0	Read PORT A
0	1	0	0	1	0	0	0	Read PORT B
X	X	0	0	1	0	0	1	Read HI ADDR
X	X	0	0	1	0	1	0	Read LO ADDR
X	X	0	0	1	0	1	1	Read EF1-4,Q
X	X	0	0	1	1	0	0	SINGLE STEP

where X denotes don't care

Table 3.2 Monitor Module Address map.

since the 1802 holds all output lines constant during the PAUSE comndition. The state codes SC0 and SC1 are strobed into the 4042 latches at TPB time. This was done in order to correlate the address and data information latched to the corresponding machine cycle. The latched values of SC0 and SC1 as well as the status of the DMAIN,DMAOUT and N0-N2 lines are read via Port B on the 8255 (configured in the input mode 0).

During the PAUSE interval between two consecutive single steps the input ports may be read by the monitor software routines running on the master PDP 11/45. Continuous stepping and readout of the Monitor enables the user to follow the microprocessor program flow as well activity on the I/O lines. The only drawback is that monitoring is not performed during real time(due to interleaved RUN and PAUSE cycles). In several instances errors in a design may arise from the presence of glitches or the failure of a response due to a fast strobe. Such errors would be difficult to detect using the stepping routine, however alternate techniques do exist for discovering such errors, such as placing the machine in a continuous loop and observing the lines with an oscilloscope, or using a logic analyzer(e,g. Biomation 920D).

The REQA line of the Controller bus is fed to the Q output ot the stepping flip flop. Basically its function is to inform the monitor software whether the the RCA CPU is in the PAUSE state. When the REQA falls low the PDP is informed that the 1802 has halted and that reading of the various ports may proceed. Essentially the PDP CPU is placed in a wait loop, continuously monitoring the state of the REQA line while the 1802 is currently running through its machine cycle.

The final function to be discussed is the actual program loading. Via Port A on the Monitor 8255 PPI the RCA object code is down line loaded from the 11/45 to the CPU module. With the 1802 in the LOAD mode, data bytes are automatically and sequentially loaded into the CPU module's on board memory. The timing representation for the load sequence is illustrated in Figure 3.10

Initially the microprocessor is RESET and placed in the LOAD mode set by a command words sent to Port C of the 8255. In the LOAD mode the 1802 is maintained in an IDLE execution state.Essentially the CPU is in a

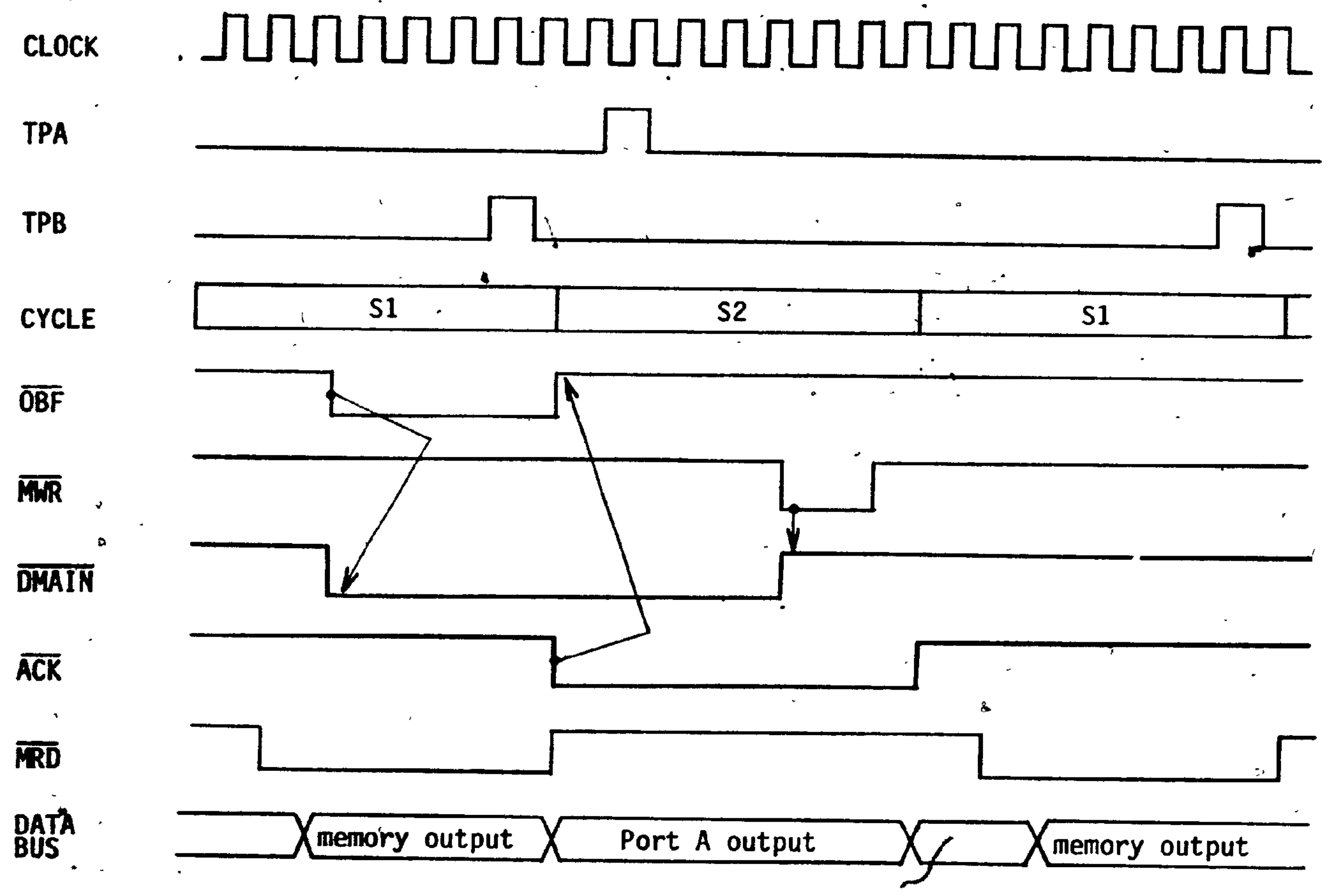


Figure 3.10 Program loading timing waveforms.

continuous S1 state and is waiting for a DMAIN request, initiated when the DMAIN line is pulled low.

With the microprocessor in IDLE, the master PDP sends out a data byte to Port A (configured in the bidirectional mode) of the 8255. Following the write sequence,the OBF line falls low indicating to the CPU module that the output buffer is full.The falling edge of the OBF line clocks the Q output of the 4013 D flip flop low thereby pulling down the DMAIN line via the hex buffer. For an S2 cycle to be executed the DMAIN line must be held low at least for the interval between TPA and TPB. Once the S2 cycle has begun,this condition is relayed back to the 8255 by decoding the SC0 and SC1 lines to drive the acknowledge ACK line input. When ACK is low the tri-state output buffers of Port A are enbabled so that Port A hs an open window to the microprocessor data bus.

With the window to Port A open the microprocessor will write the data on the bus into memory location pointed to by its program counter (R0). This occurs during the time interval when the MWR line of the 1802 is low. Furthermore the falling edge of MWR is used to preset the flip flop thereby removing the DMAIN request line. It is important that only one S2 cycle is executed for each output byte from the PDP; had the DMAIN line not been removed another S2 cycle would have been executed and consequently the same byte from Port A would have been loaded twice into contiguous memory locations. After completion of the S2 cycle the machine returns to its IDLE execution state until the next DMAIN request. Note that throughout the entire loading process the WAIT and CLEAR lines are held in the LOAD condition.

The REQ B line tied to the DMAIN line is used to inform the PDP about the status of the loading sequence. Whem REQ B is negated it informs the PDP that the transmitted byte has been entered into Port A and that a DMAIN cycle has been requested from the microprocessor. The low state of the REQ B line disables the Monitor software from sending the next byte until the present byte has been read into the CPU memory.On completion of the S2 cycle the ACK line and hence the window into Port A is removed.

Normally the TPA clock is disabled during IDLE states however is enabled by the microprocesor during an S2 cycle. Furthermore TPB is used to strobe the data bus of the CPU data bus into Port A; hence during the interval S2 and TPB, Port A is both in the input and output conditions. In order to disallow such contention TPB is disabled during a DMAIN cycle by the NAND operation with the S2 state.

An interesting feature of the loading sequence is that on the following S1 cycle the byte that has just been stored in the CPU memory is loaded back into Port A by virtue of the TPB strobe. As Port A is a R/W port the same byte is read into the PDP by the Monitor software and compared logically with the byte just transmitted. Although the read and compare sequence slows down loading, the technique allows for detection of transmission errors. Should an error occur the user is notified by the monitor software and another loading attempt is required.

The Monitor module is packaged identically as the CPU module on a double sided 4.5"X8" PC board. CMOS components are used extensively for power consumption reduction.

3.7 SUMMARY

Tying into the PDP via the DR-11C allowed effective access to the PDP Unibus and interface with the 11/45 system software. The Controller satisfied the input and output needs of the design,yet left ample room for improvements and enhancements.

At the time of the design the RCA 1802 microprocessor offered architectural and power requirement advantages over other available processors. It appears that with second sourcing and speed improvements due, the device will maintain its availability for several years. Incorporation of the 1802 micrprocessor into the the CPU module offered a cost effective,powerful and compact computing package. Linkage of the CPU module and the Minicomputer provided by the Monitor module gave adequate control and functionality for prototype development. The main feature is highlighted by the fact that linkage is totally transparent to the overall system.

CHAPTER 4

SOFTWARE DESCRIPTION

4.1 INTRODUCTION

Control, monitoring and program loading is handled by software programs from the master minicomputer. Software modules for the various operations are written in PDP 11/45 Assembler, the protocol predefined in conjunction with the implemented hardware design. The prototype software is presented in this chapter; simple in structure these basic routines allowed for testing of the hardware in real time execution and offer a more complete understanding of the hardware/software interaction.

4.2 DR-11C PARALLEL I/O

Transfer of data and status to and from the Controller is accomplished via the the three buffers/registers of ther DR-11C general purpose interface. The three registers are defined by their absolute address as follows;

DRIN=167764

DROUT=167762

DRCSR=167760

Here and in all following discussions all absolute values are specified in octal base. In using an I/O memory mapping scheme the PDP CPU treats the DR11C registers as memory thereby facilitating the use of these registers within the software.

The DRIN is a read only buffer for data transfers from the Controller to the DR-11C. An example of an input instruction would be MOV DRIN,R1 in which the the data from the DRIN buffer is moved to R1,an internal PDP register.

The DROUT is a read/write register which handles data transfer from the DR-11C to the Controller. A typical output instruction is MOV (R1),DROUT in which the contents of the address pointed to by R1 is moved to the DROUT.

The DRCSR is a 6 bit read/write register for two way transfer of status and control between the DR-11C and the I/O Controller,

4.3 DATA/ADDRESS MULTIPLEXING

Data and address words are multiplexed onto the common bidirectional bus of the I/O Controller. The proper interpretation of the information sent from the DROUT as being either data or an address is made by appropriately decoding the CSR0 and CSR1 bits of the DRCSR. Depending on how the CSR bits are set by software determines whether an ADDR or DATO strobe is issued. As an example an instruction sequence for transmitting the data byte 55 to the address 342 would be as follows;

```
MOV #0,DRCSR        ;ADDRESS MODE
MOV #342,DROUT      ;TRANSMIT,ADDRESS
MOV #1,DRCSR        ;DATA MODE
MOV #55,DROUT       ;TRANSMIT DATA
```

4.4 8255 INITIALIZATION

Transfer of data and control to the microprocessor is done via the 8255 Programmable Peripheral Interface chip on the Monitor module. Initially the 8255 must be set in its operational mode accomplished by loading the appropriate control words. The required sequence is as follows;

```
MOV #0,DRCSR
MOV#310,DRCSR        ;SELECT 8255
MOV#1,DRCSR
MOV#302,DROUT        ;8255 MODE
MOV#15,DROUT         ;SET INTE1
MOV#12,DROUT         ;SET INTE2
```

Notice that once the bus is dedicated to data transmission (MOV#1,DRCSR) it remains in the same mode until changed. Similarly once an address is selected it remains so until another address is issued.

4.5 MICROPROCESSOR MODE CONTROL

With the 8255 initialized, data may be transmitted or read to any of the various ports on the chip. The four operational modes of the microprocessor (LOAD,RESET,RUN,PAUSE) are controlled through the WAIT and CLEAR lines via port C bits 1 and 0 (PC1,PC0) of the 8255. As an example the following instruction sequence is used to set the 1802 microproceesor in the LOAD mode;

```
MOV#0,DRCSR
MOV#210,DROUT        ;SELECT PORT C
MOV#1,DRCSR
MOV#1,DROUT          ;1802 RESET
MOV#0,DROUT          ;1802 LOAD
```

4.6 PROGRAM LOADING

Data to and from the microprocessor is channelled through Port A of the 8255. Loading of a program into th CPU module is done sequentially using the LOAD mode of the microprocessor. Presently, on board memory allows for 1024 bytes of RAM storage. The program loading flowchart is given in Figure 4.1. Note that the assertion of the REQB flag (bit 15 DRCSR) enables the software to output the next data byte.

In the following example data bytes 63 and 64 are loaded consecutively into memory.

```
        MOV#0,DRCSR
        MOV#10,DROUT        ;SELECT PORT A
        MOV#1,DRCSR
        MOV#63,DROUT        ;OUTPUT DATA
WAIT:   TST DRCSR           ;TEST REQ B
        BPL WAIT            ;WAIT IF REQ B LOW
        MOV#64,DROUT        ;OUTPUT DATA
```

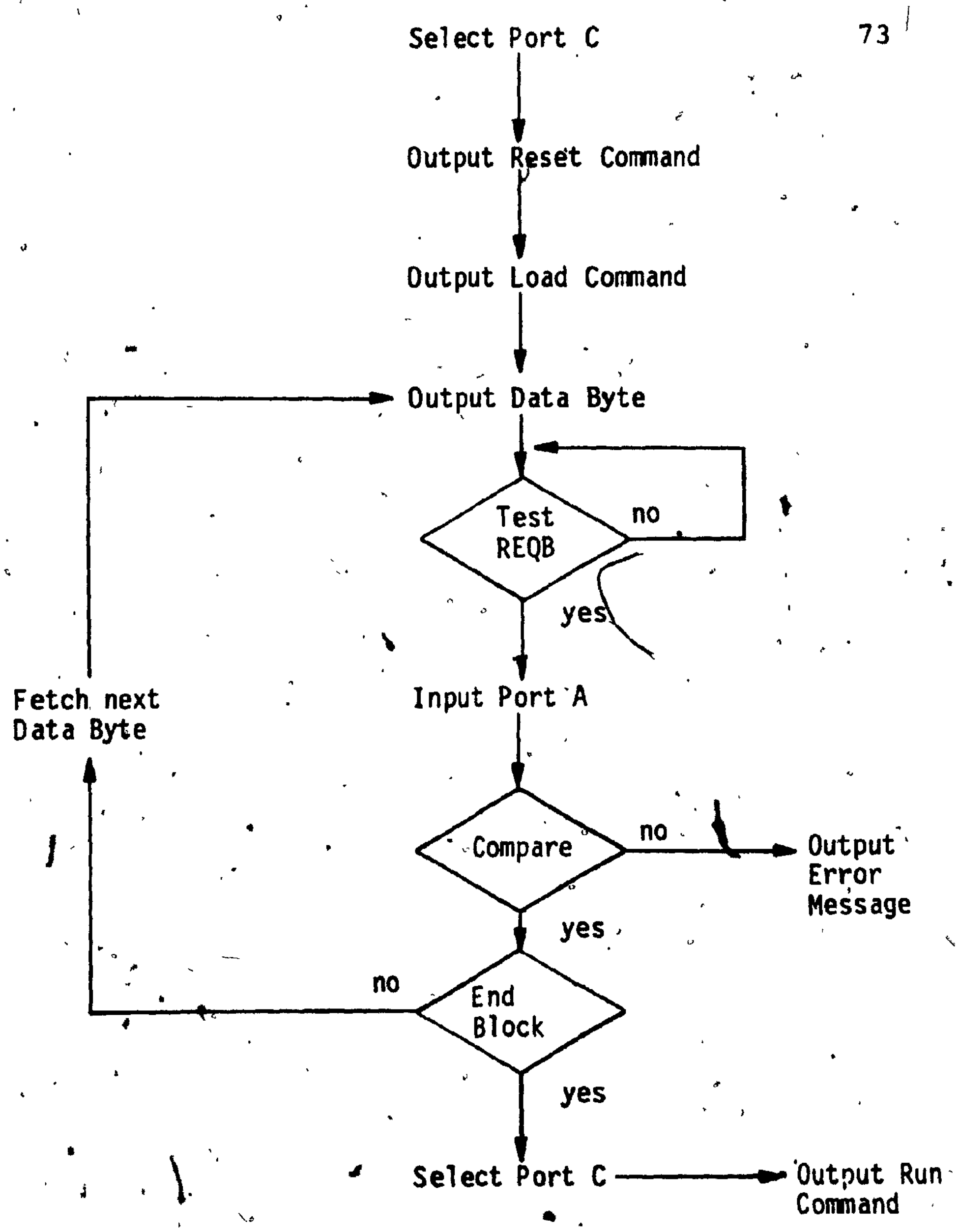


Figure 4.1 Program loading flow chart.

Loading is handled automatiqally by the 1802 LOAD mode and the implemented hardware. REQ B is tested after each data transfer and only when asserted high may the next data byte be transferred.

4.7 SINGLE STEPPING

Single stepping of the microprocessor allows for a program sequence to be executed in a pseudorun manner. Single stepping occurs by alternately enabling the 1802 to run through single machine cycles. Stepping is handled by the Monitor module hardware but is always initiated by a software command. Initially the 1802 must be set in the PAUSE mode; a single step is performed each time the SINGLE STEP address is issued and allows for a single machine cycle to run. The sequence may be coded as follows:

```
        MOV#0,DRCSR           ;
        MOV#14,DROUT          ;1802 STEPPED
WAIT:   TESTB DRCSR           ;TEST REQ A
        BMI WAIT              ;WAIT IF REQ A HI
```

Following a single step the microprocessor enters the PAUSE mode indicated by the REQ A asserted low. By testing the condition of the REQ A the master software is informed of the CPU state and may perform a data input sequence and/or continue single stepping,

4.8 INPUT PORTS

The data-in sequence may be used to read data from any of five input ports on the Monitor module. Each port latches relevant information from the microprocessor I/O lines. The five port contents and their absolute addresses were given in Table 3.2. A typical read sequence from the Monitor module may be as follows:

```
MOV#0,DRCSR
MOV#10,DROUT          ;SELECT PORT A
MOV DRIN,R0           ;READ DRIN INTO R0
```

During the PAUSE period the master software may read the information from the Monitor module ports and consequently search for a specific break point and/or display the information to the user on an output device(CRT). Figure 4.2 illustrates a simplified flowchart for continuous single stepping, with line monitoring and break point functions.

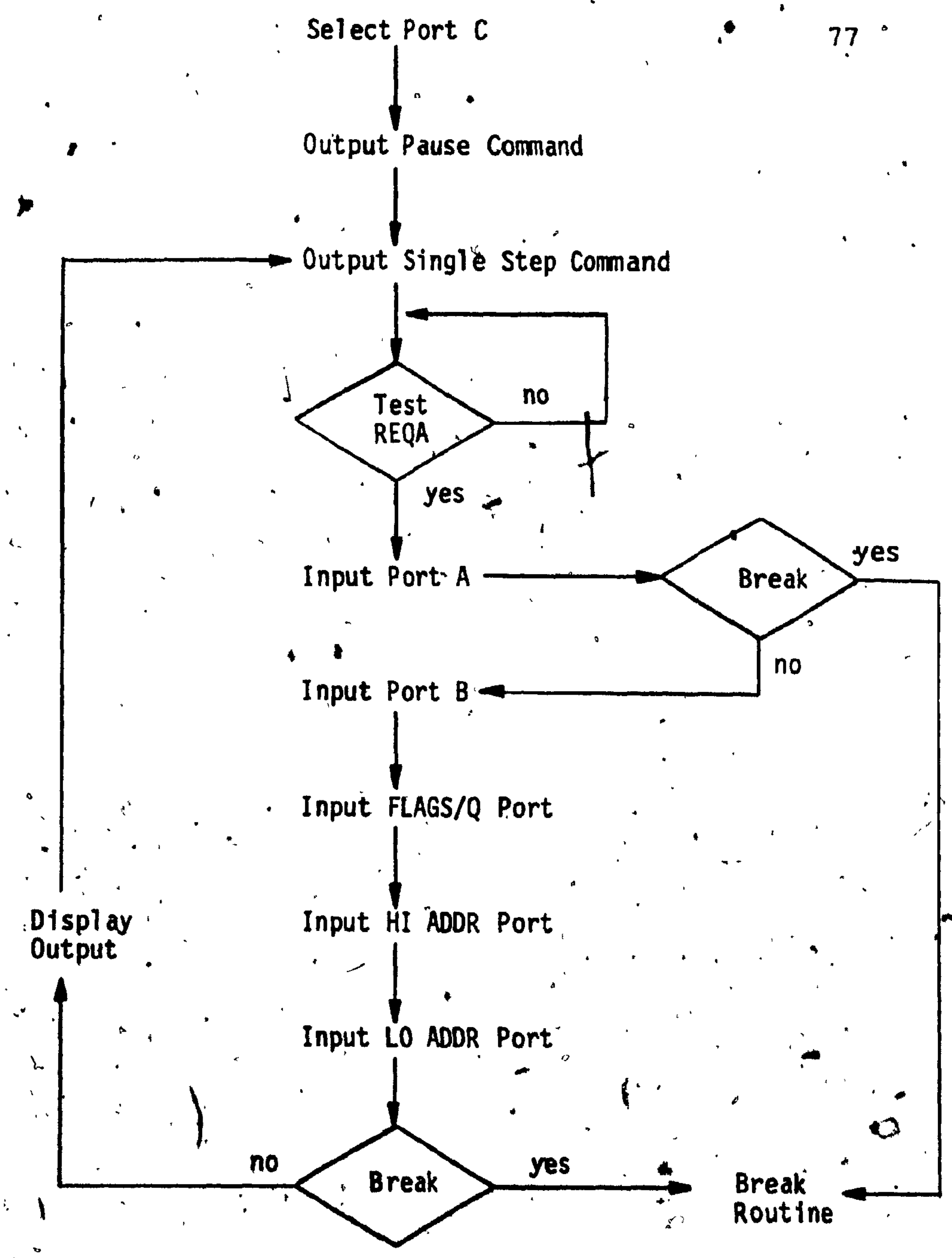


Figure 4.2 Single stepping, break point search flow chart.

4.9 SUMMARY

Using MACRO-11 software the neccessary protocols for control, monitoring and loading functions for the monitor module have been implemented into separate modules. By interfacing directly with the hardware these modules form the direct link between the development system software and hardware. In essence the modules form the lower layer of the development system software package; the upper layers or supervisory software need only to reference these modules in terms of subroutine calls or macros. Modularity and simplicity in structure allowed easy interfacing between software and hardware, thereby effectively reducing development time.

CHAPTER 5
SYSTEM IMPLEMENTATION

5.1 INTRODUCTION

The preceeding chapters have given a detailed description of the various hardware and software module elements of the target development system. We now proceed with a generalized description of the use of these modules in an 1802 microprocessor development package and its use in a particular application. The application called for the design of a microprocessor based waveform analyzer to be used in a real time industrial environment. Complete performance specifications of the development system software and and the application may be found in the references(4,6,7).

5.2 DESIGN TARGET

The target design comprises of a microprocessor controlled portable desk top instrument. The instrument's electronics were broken down into functional analog and digital subsystems; i.e., CPU module, A/D and D/A modules, keyboard controller, etc. Each subsystem was designed as an individual hardware module along with its associated software driver module. Such modularity was absolutely neccessary for testing and debugging, field support, and flexibility for re-use and enhancements.

The intellegence of the system is contained in the CPU module. Data, address and control timing of the other sub-systems were designed with respect to the CPU module. An extra slot in the card cage was alloted for the monitor module. Integrating the monitor module during development allowed for down loading programs and testing hardware and software interactions. Its presence was neccessary only for initial development and test and was totally transparent to the overall system operation.

5.3 DEVELOPMENT SYSTEM SOFTWARE

Using standard RT-11 software, application programs written in the RCA microprocessor assembly language (COSMAC) are editted, assembled,linked and loaded by the PDP 11-45 minicomputer. Source programs are prepared in the COSMAC instruction set using the RT-11 editor. Thus in developing an application microprocessor program the user has access to the full capabilities of the RT-11 editor resources. Source application programs may be stored and accessed from the RK05 disk operating system. In comparison with other storage mediums found on several development sytems it is apparent that the hard cartridge disk is a favorable alternatrive in terms of cost/bit and speed. A cross assembler (RCA.SML) performs the translation of the RCA assembly language into PDP 11-45 assembly by defining the complete COSMAC instruction set in terms of macros. The PDP 11 like most minicomputers, has a macro asembler. Macros allow the user to define a desired sequence of instructions by a particular user defined name. During the machine assembly of a program, each time the macro name is encountered the assembler will automatically insert the instruction set defined by the macro name. Using this technique to assemble RCA (or any other microprocessor language) instruction set, a macro definition of each instruction is made which instructs the PDP assembler how to construct the RCA machine code for that instruction. Thus MACRO 11 (the PDP 11 assembler) is enabled to act as the COSMAC assembler; whatmore the MACRO 11 assembler becomes a virtual assembler for any other microprocessor. Due to the interactive nature and ease of cross-assembler development, the capabilities for implementation to almost any microprocessor makes this method both flexible and very cost effective.

The cross assembler is an important aspect of the development system, however the prospective user need not be concerned with the details of assembly, but merely requires the basis of operating it. Using the cross-assmbler, complexity of application software is no drawback to final asembly. Furthermore due to the nature of the minicomputer CPU and disk controller, assembly times are relatively fast thereby reducing development times and costs.

The concept of macro generation may be extended by allowing for the generation of COSMAC instruction macros. By setting up nested macro routines, during assembly time, a given macro may further be defined by another set of macros.This option of macro generation in the microprocessor application software is a particularly useful feature for program development. Common instruction sets may be defined once, thereby easing program writing. Furthermore, commonly used macros may be linked with the cross assembler for future use, and may easily be added or deleted at will by the programmer.

Once assembled the program is ready for loading and testing. The main monitor program (MONIT) is resposible for the down line loading of the application program and provides an on line real time execution of the program in the prototype . Program execution may proceed in either continuous mode (real time execution) or single stepping. In the single step mode the various lines of the microprocesor are read by the PDP minicomputer. Break points at the occurance of a data or address byte is accomplished by repeated single stepping and monitoring of the address or data lines. The sofware modules for the various functions are part of a MACRO-11 program RCALOAD. The actual monitor program MONIT is written in standard FORTRAN and accesses the various modules within RCALOAD through subroutine calls. Programs MONIT and RCALOAD are combined during link time to produce the neccessary core image executable module. User-machine interaction is channelled through the DEC-writter and GT 50 graphic terminal.

The monitor program is called directly from the PDP system monitor. Once initiated the monitor program is entirely intrecative; that is the program directs the user in loading programs, requsts commands and may be interrupted at any time by the user. Exiting from the monitor returns control once again to the PDP system monitor.

In order to load a program the user must specify the name of the object module (residing on disk) and the length of the program in terms of pages (256 bytes).During loading from the disk file into the prototype memory, transmission errors are continuously checked for, providing a guarantee that the proper program has been loaded. If a transmission error does occur, a PARITY ERROR statement is output to the terminal. Under such conditions the user would attempt a reload.

Once loaded the monitor program offers the user various modes of operation; these include NEW,RESET, RUN,STEP,LOAD,ADDR and DATA. Under RUN the loaded program is run in real time. The user may find it useful to observe various timing activities with either an oscilloscope or logic analyzer.

Entering the STEP mode the microprocessor is continuously stepped through one machine cycle at a time. Each cycle is displayed as a single line on the output terminal in scroll fashion during single step execution. The address lines,data lines,program pointer, mnemonics, along with the state of the machine and control lines are displayed; thereby allowing the user a detailed obervation of the machine state at any time throughout program execution.

The ADDR and DATA commands are incorporated for the setting of a break point at the occurance of a particular address or data. In searching for the break point the program sequences (psuedorun speed) the microprocessor until a match is made. During the search the graphic output is disabled in order to speed up the search process. Both these commands are very useful, especially the ADDR command in following program flow through subroutines and interrupts.

The RESET command causes a dead halt and reset of the microporcessor. Finally, The NEW command allows the user to load a new file from disk and proceed with monitoring functions.

5.4 SUMMARY

An application called for the design of a microprocessor controlled portable instrument. The intelligence required for the processing of data and handling I/O used the resources of the CPU module. The monitor module allowed linkage of the prototype to the development system.

Using the function software modules(defined in Chapter 4) an effective development system package has been developed. Using RT-11 software COSMAC application programs are editted, assembled, linked and loaded by the PDP 11/45 minicomputer.

This system offered a vital tool for the development of the neccessary application hardware . Furthermore it offers the user the full resources of the RT11 operating system.

CHAPTER 6
CONCLUSIONS

Confronted with a wide range of development systems the concept of configuring a system around a minicomputer is attractive for several reasons. The built in flexibility makes the system adaptable to any 8 or 16 bit microprocessors, thus the system is upgradable as new devices appear. The user is not forced to reinvest thousands of dollars each time development for a new processor is desired. Furthermore, the operating system of the minicomputer offers performance and speed capabilities that cannot be matched by any of the current off-the-shelf development systems.

We have proposed one possible configuration revolving around a PDP 11/45 minicomputer and RCA 1802 microprocesor. An important achievement of the overall system has been modularity. CPU and Monitor PC boards have been designed for easy prototype integration. Additional plug in modules for various support functions and I/O may similarily be configured. The same principle holds for commonly used software routines which are linked into the final development system routines. On conversion to varying processors only the lower software layers, i.e.,those interfacing directly with the hardware, need be reconfigured while retaining the supervisory upper layers.

Performance figures and comparisons among microprocessor development systems are difficult to compile. One can only judge ones own experience with a given system and compare with others. In terms of our design performance, from the initial design of the Controller to the time the final prototype was delivered, involved approximately a 1 man-year period. Although additional effort was required in getting the system operational, in the long run we feel the approach we have taken has outweighed that of acquisition of a ready made package.

REFERENCES

1. H.A.Cohen and R.S.Francis,"Macro-Assemblers and Macro-Based Languages in Microprocessor Software Development," Computer, Feb.1979, pp.53-64.

2. R.Martinez,"A Look at Trends in Microprocessor/ Microcomputer Software Systems, "Computer Design, June 1975, pp.51-57.

3. L.Krummel and G.Schultz,"Advances in Microcomputer Development Systems," Computer, Feb. 1977, pp.13-19.

4. R.Beyar,"Microprocessor Assembly and Simulation Using MACRO-11, "Proc. DECUS Banff Conf.,1977, pp.811-815.

5. C.L.M.Stoute, "PDP-11V03 As A Universal Development System, "Proc. DECUS Banff Conf., 1977, pp.795-804.

6. R.Beyar, J.Kishon, S.Gracovetsky and R.Moses, "Microprocessor Cross-Assembler and Simulator Construction Using MACRO-11 Macro Libraries," Proc. MIMI 77, Nov. 1977, pp. 101-105.

7. J.Kishon, S.Gracovetsky, R.Moses and R.Beyar, "Analysis of Signals In The Time Domain," Proc. MIMI 77, Nov. 1977, pp.255-260.

8. Digital Equipment Corporation, "DR-11C General Devices Interface Manual," Maynard, Mass., June 1974.

9. RCA Solid State Division, "User Manual for the CDP 1802 COSMAC Microprocessor," Somerville, N.J., 1976.

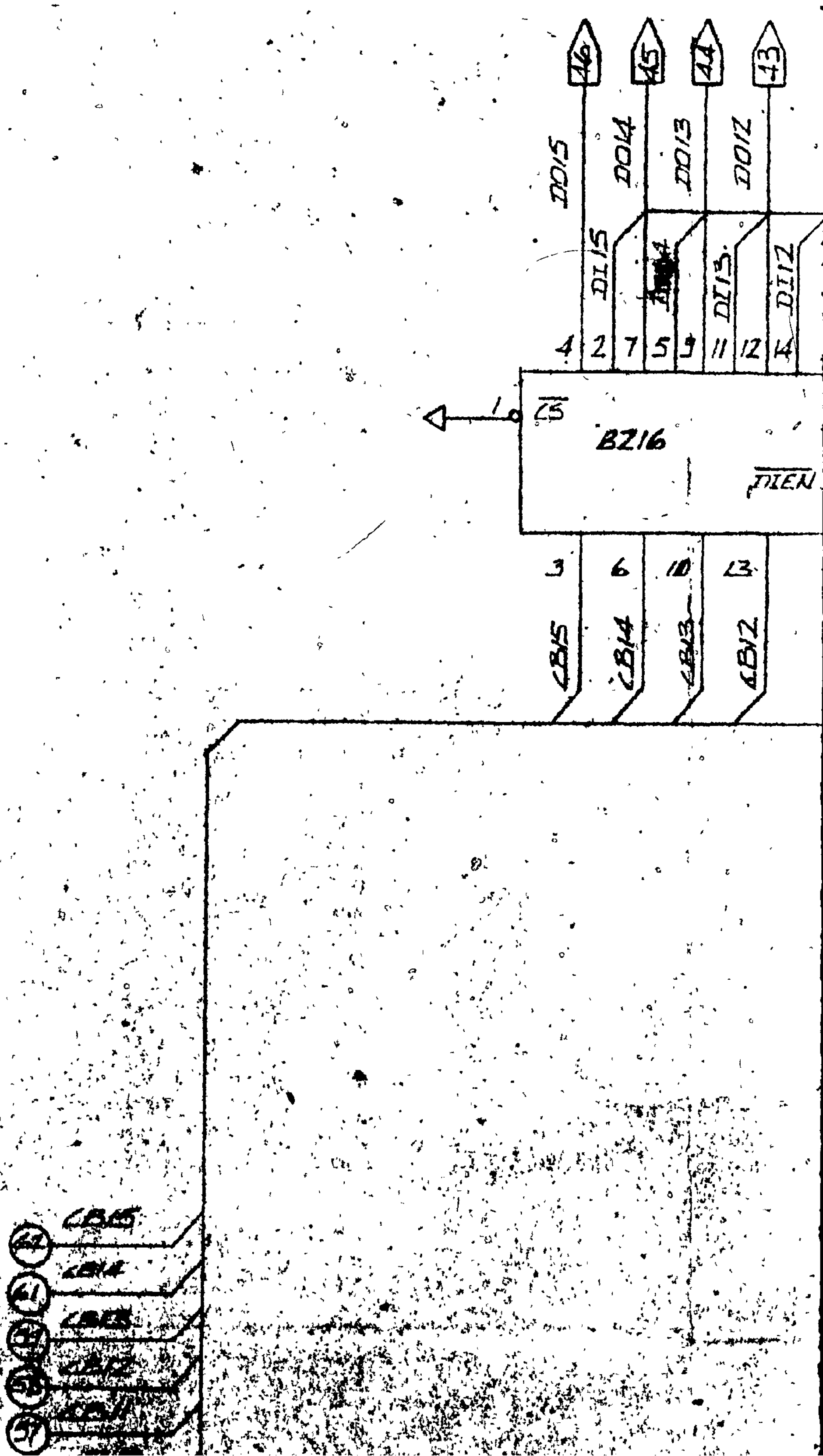
10. B.J.Carey, "A Microprocessor Laboratory For A University Environment," Computer, Jan. 1977, pp. 40-46.

11. R.Francis and R.Teitzel, "Real-Time Prototype Analysis As A Microprocessor Design Aid," Computer Desing, Dec. 1978, pp. 65-73

12. J.L.Pokoski and O.Holt, "Developing Software For Microcomputer Application," Computer Design, Mar.1975 pp.88-90,

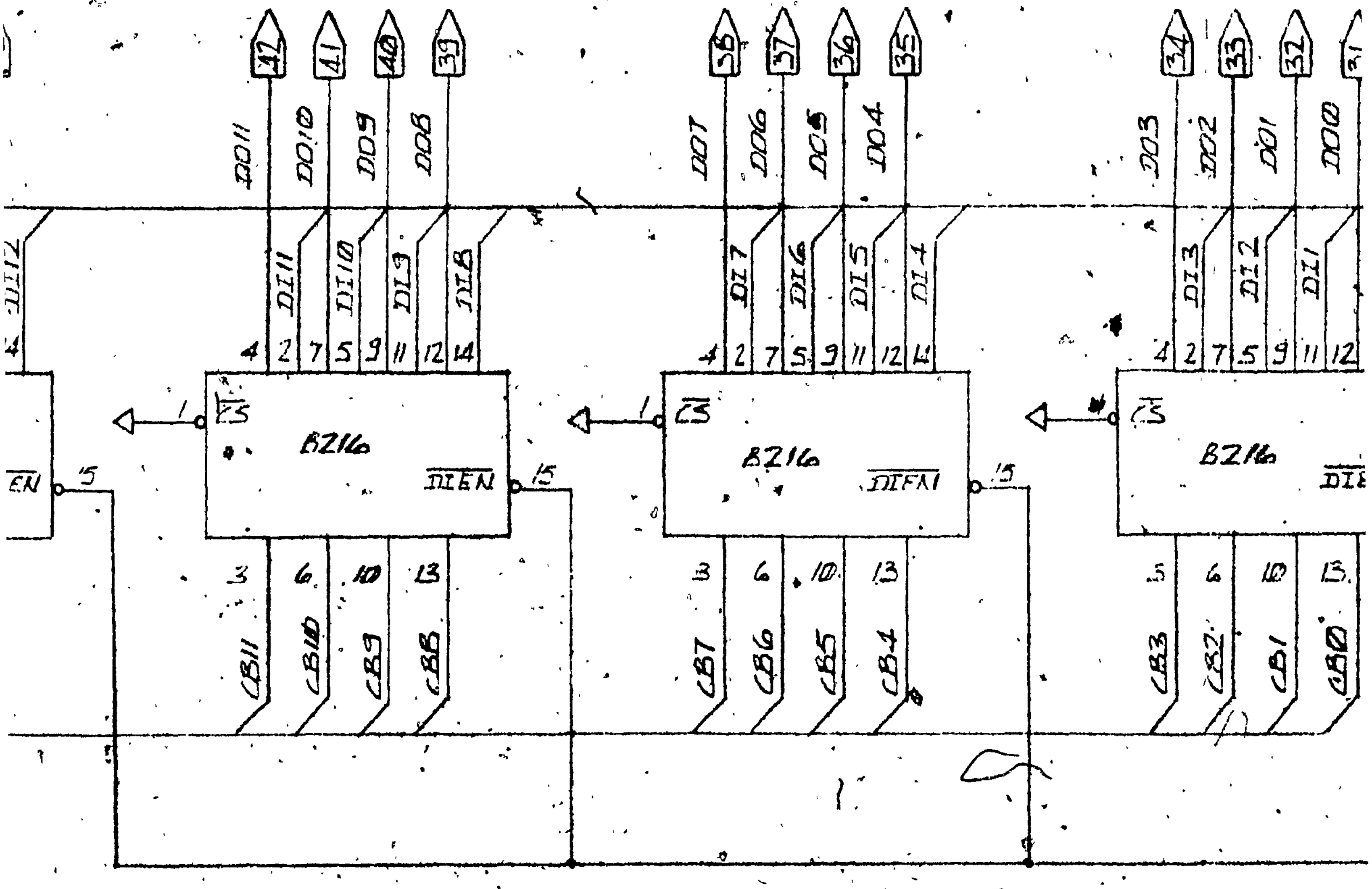
APPENDIX

System Schematics


46
45
44
43
DO15
DO14
DO13
DO12
DI15
DI13
DI12
4
2
7
5
9
11
12
14
1
CS
BZ16
DIEN
3
6
10
13
CB15
CB14
CB13
CB12
62
61
CB15
CB14

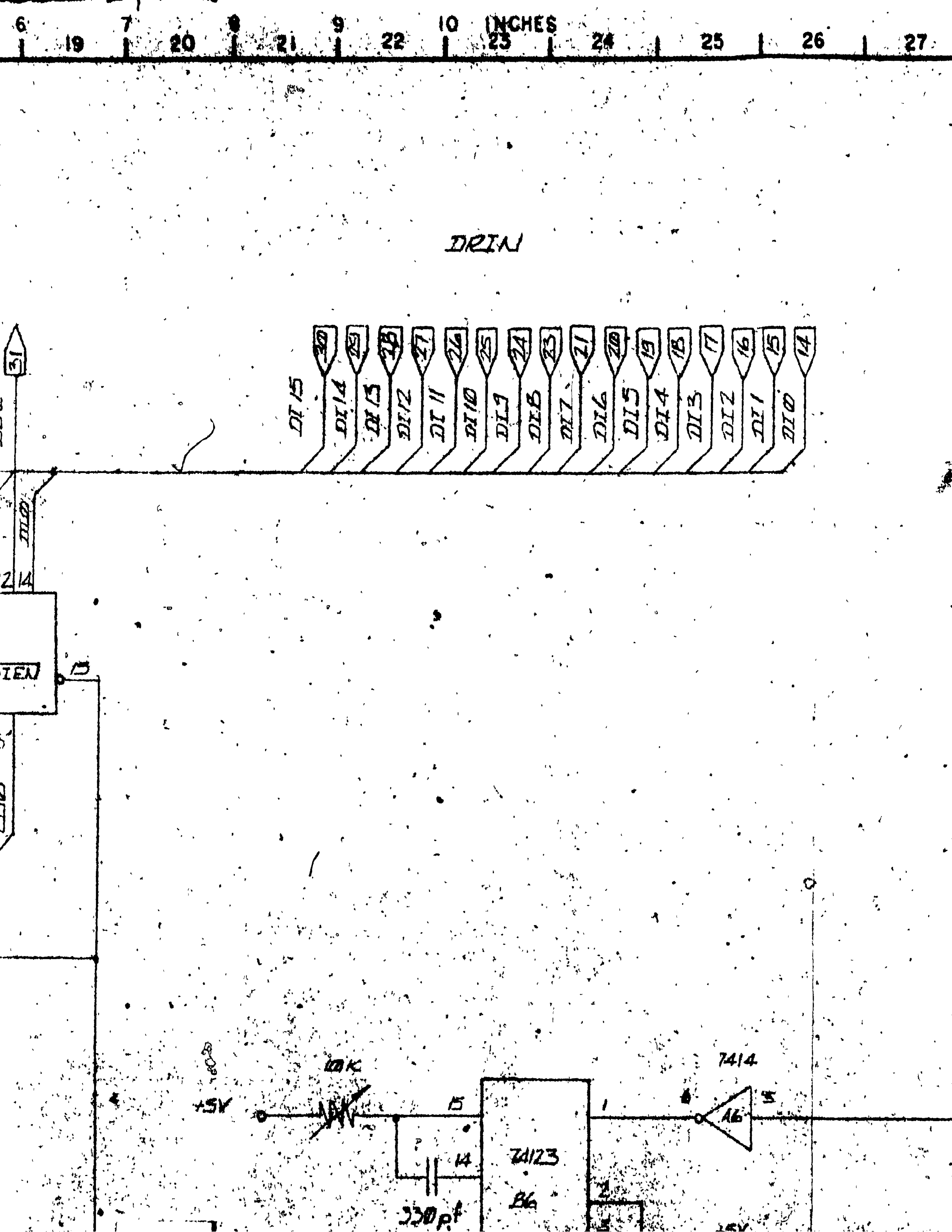
1 of

DROUT



42
41
40
39
38
37
36
35
34
33
32
DO11
DO10
DO9
DO8
DO7
DO6
DO5
DO4
DO3
DO2
DO1
DO0
DI12
DI11
DI10
DI9
DI8
DI7
DI6
DI5
DI4
DI3
DI2
DI1
CS
8216
DIEN
CB11
CB10
CB9
CB8
CB7
CB6
CB5
CB4
CB3
CB2
CB1
CB0

24



6 19 7 20 8 21 9 22 10 INCHES 23 24 25 26 27
DRIN
30 29 28 27 26 25 24 23 21 20 19 18 17 16 15 14
DI15 DI14 DI13 DI12 DI11 DI10 DI9 DI8 DI7 DI6 DI5 DI4 DI3 DI2 DI1 DI0
31
DI0
14
DIEN
15
+5V
100K
15
14
330pf
74123
B6
1
2
7414
A6
5

26 27 28 29 30 31 32 33 34

Third Angle Projection

Dimensions In Inches ☐ Mms. ☐

Scale

Tolerances

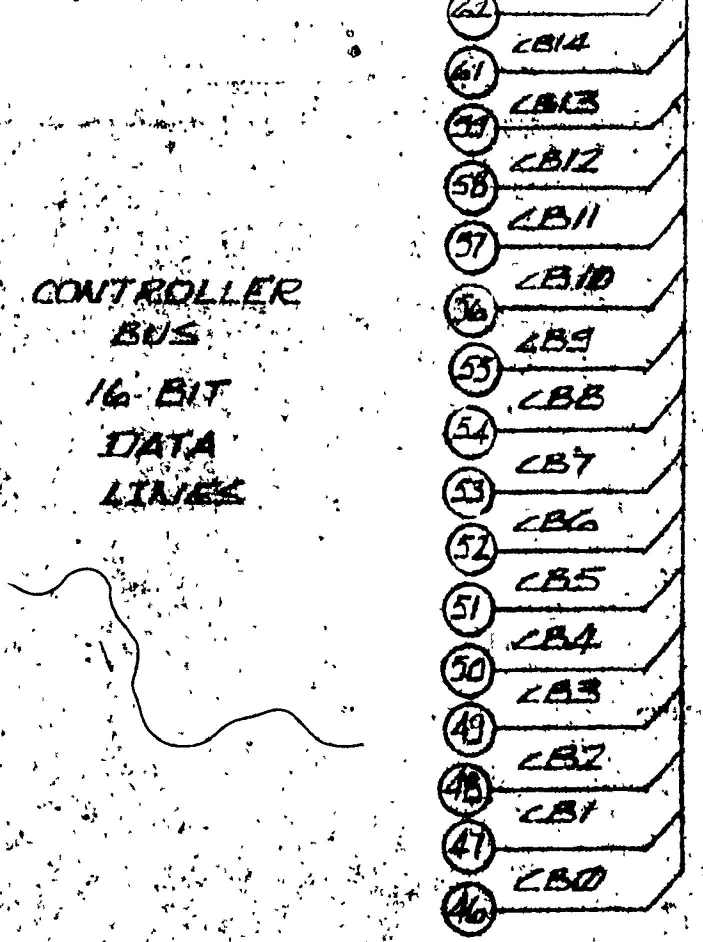
Material

Finish

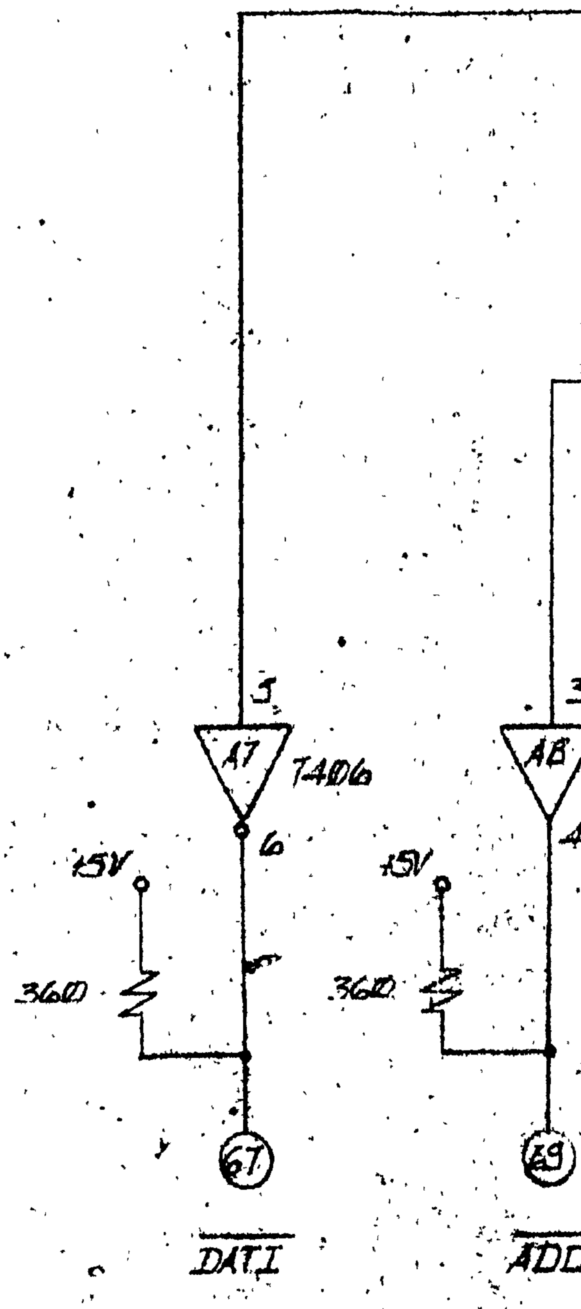
Revisions

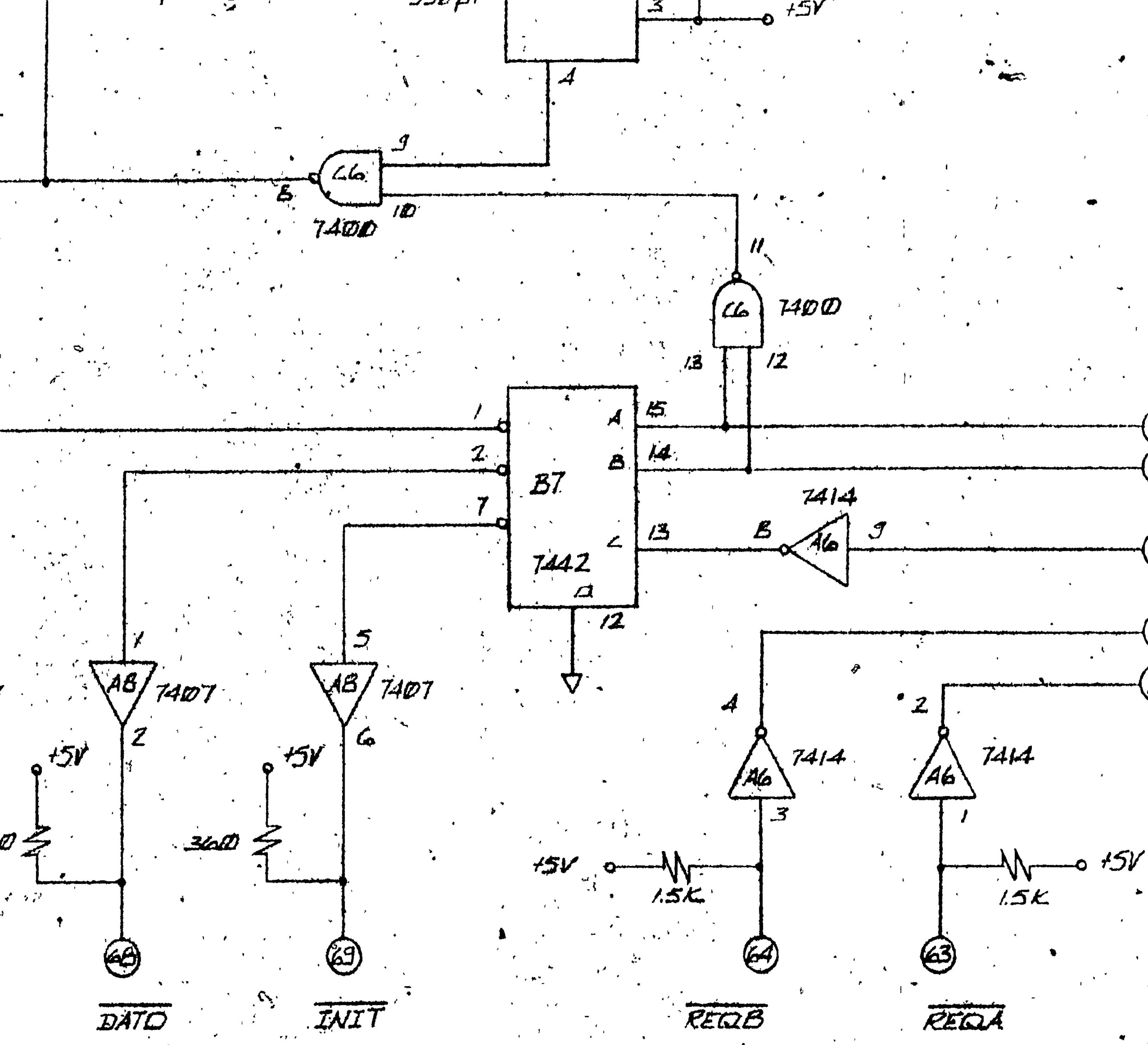
Iss	FRO DCO	App.	Rel.	Mic.

A B C D E F G H J K


CONTROLLER BUS
16 BIT DATA LINES
62
CB14
61
CB13
59
CB12
58
CB11
57
CB10
56
CB9
55
CB8
54
CB7
53
CB6
52
CB5
51
CB4
50
CB3
49
CB2
48
CB1
47
CB0
46

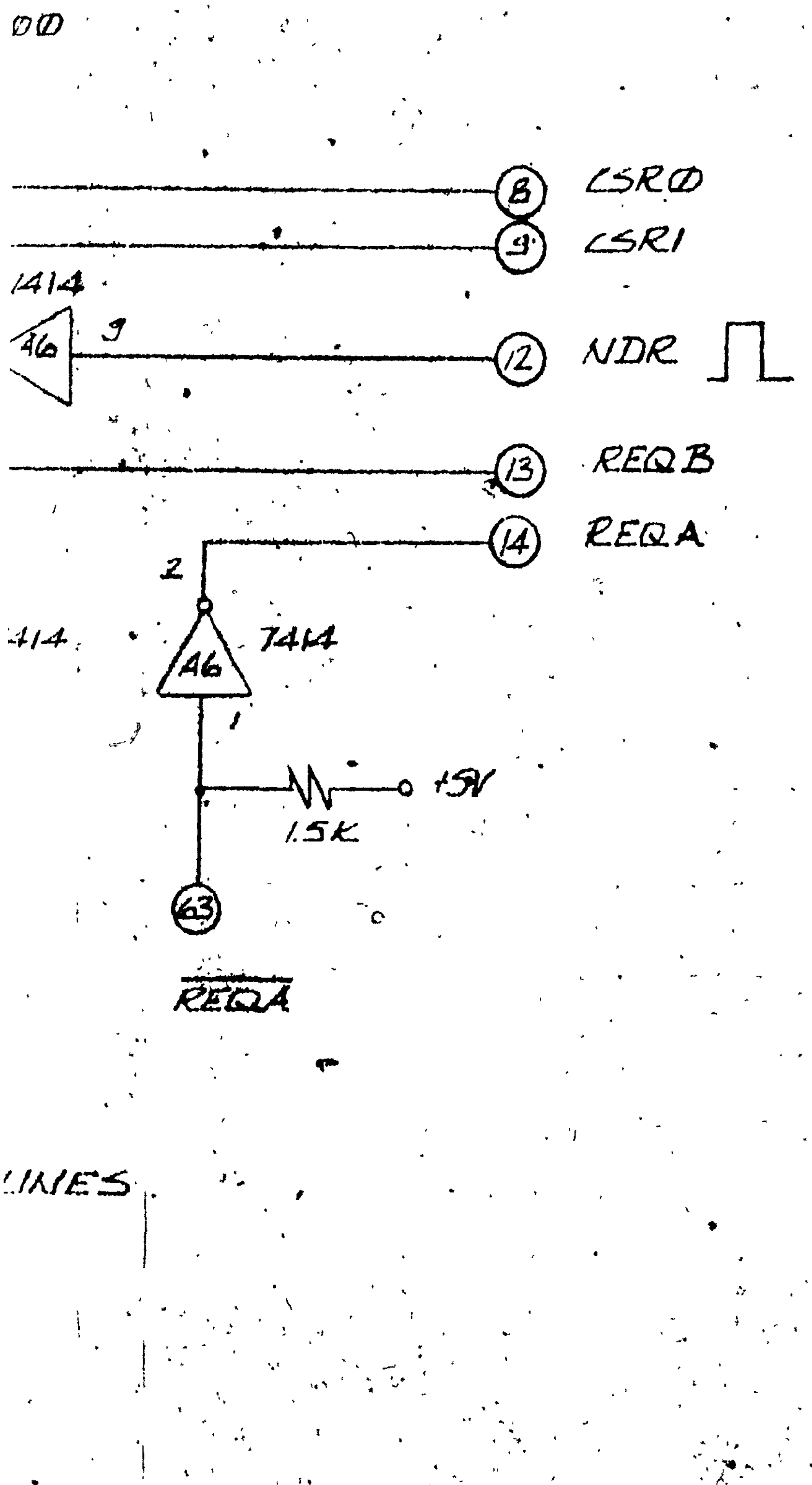
5 of


3
A7
7406
6
+5V
360
67
$\overline{DATI}$
3
A8
4
+5V
360
59
$\overline{ADD}$



CONTROLLER BUS CONTROL LINES

7 of



Designed	Drawn	Checked
Approved [illegible]	Approved	Approved
Drafting App.	Release DEC. 7 1977	
Reference		
Next Ass'y or Family Tree		

AES

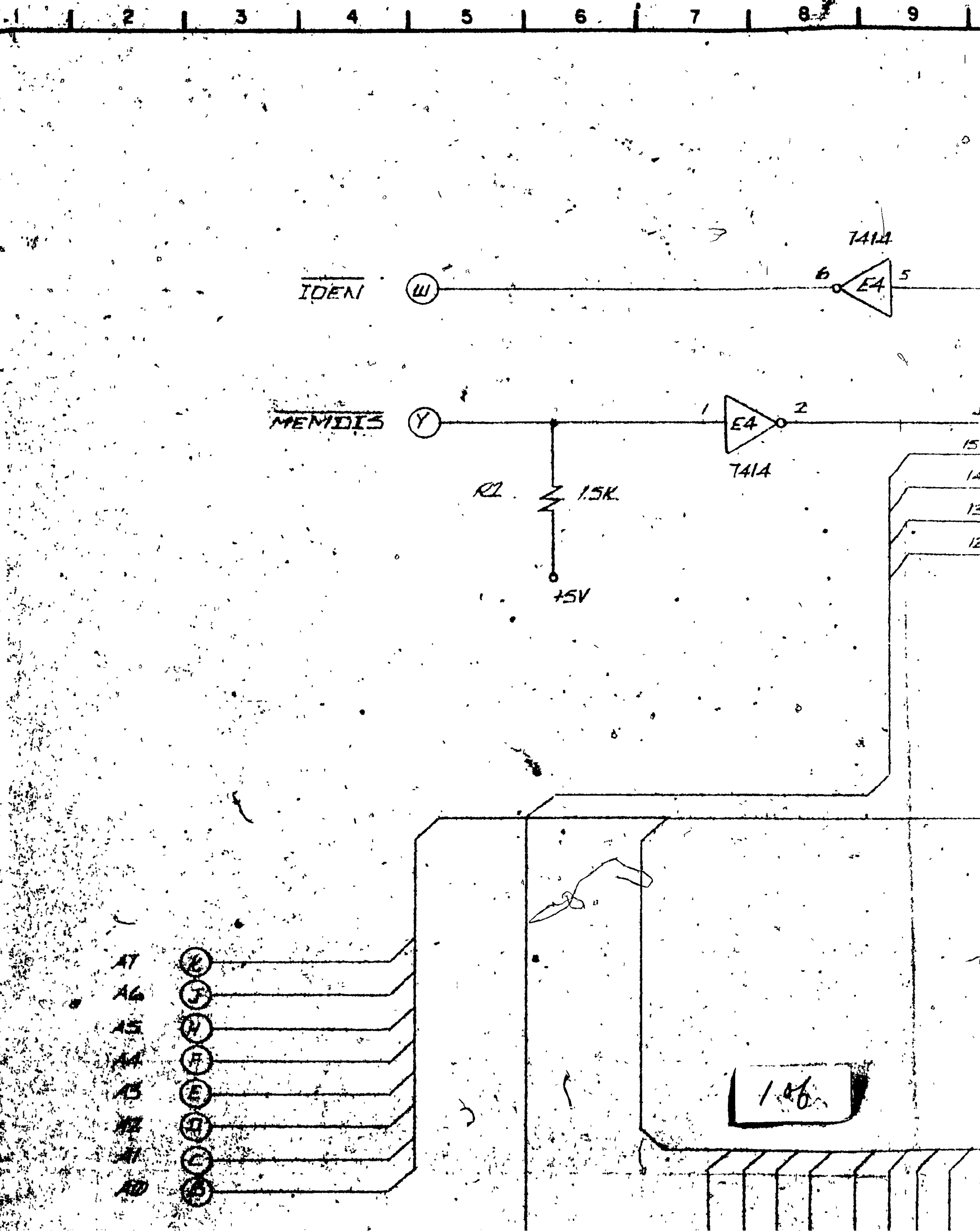
AES Data Ltée/Ltd

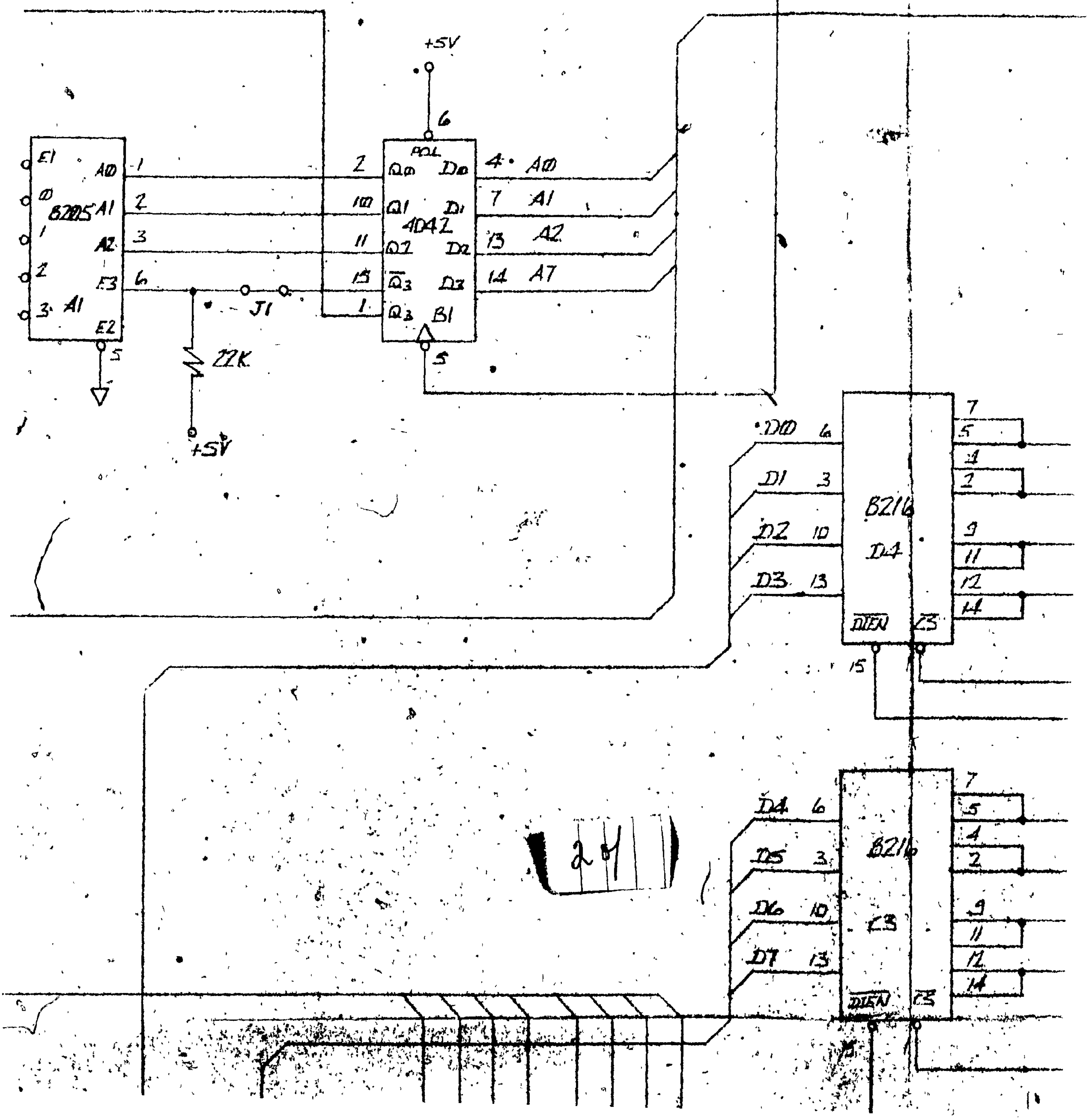
Title

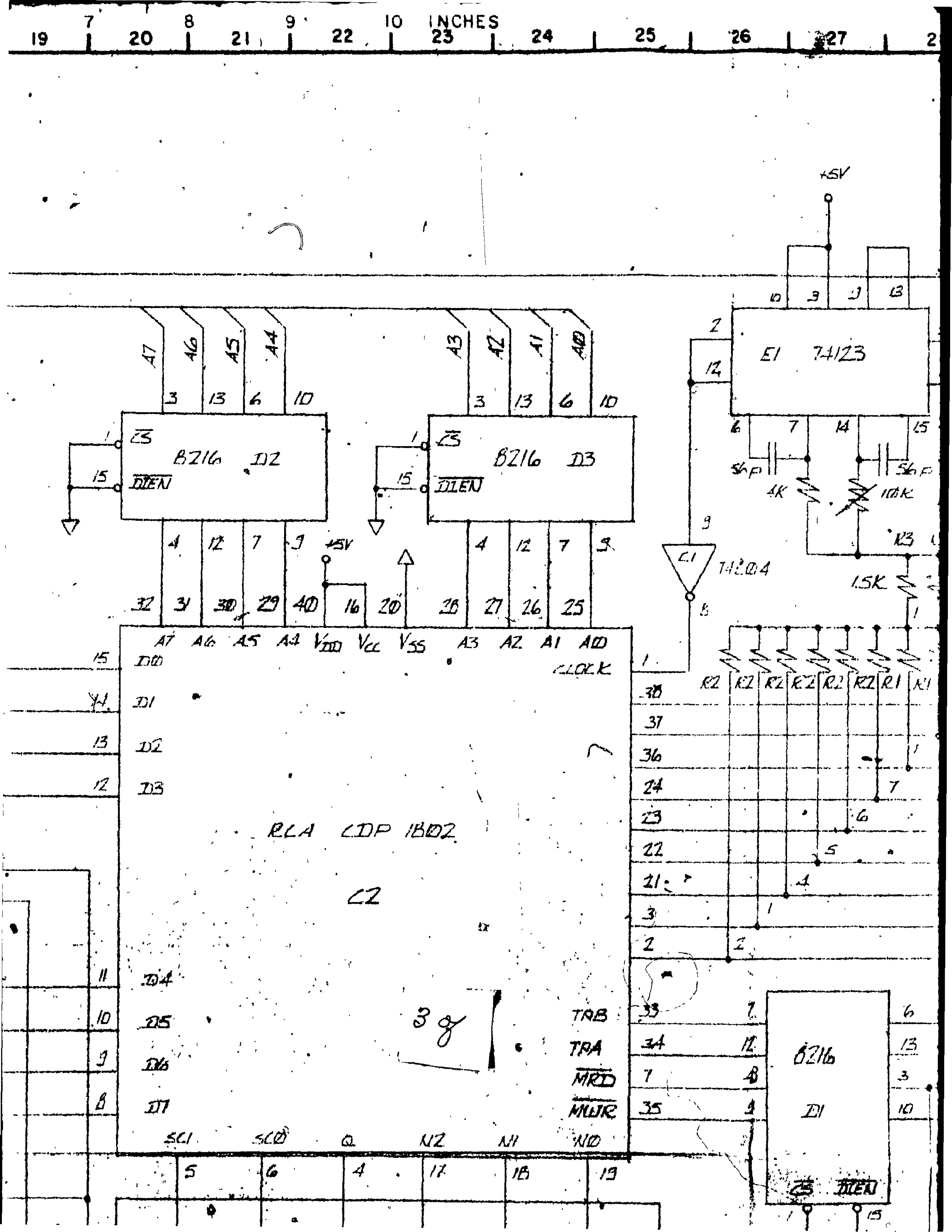
CONTROLLER

Size D	Drawing No.
Issue 02	Sheet 1 of 3 Sheets

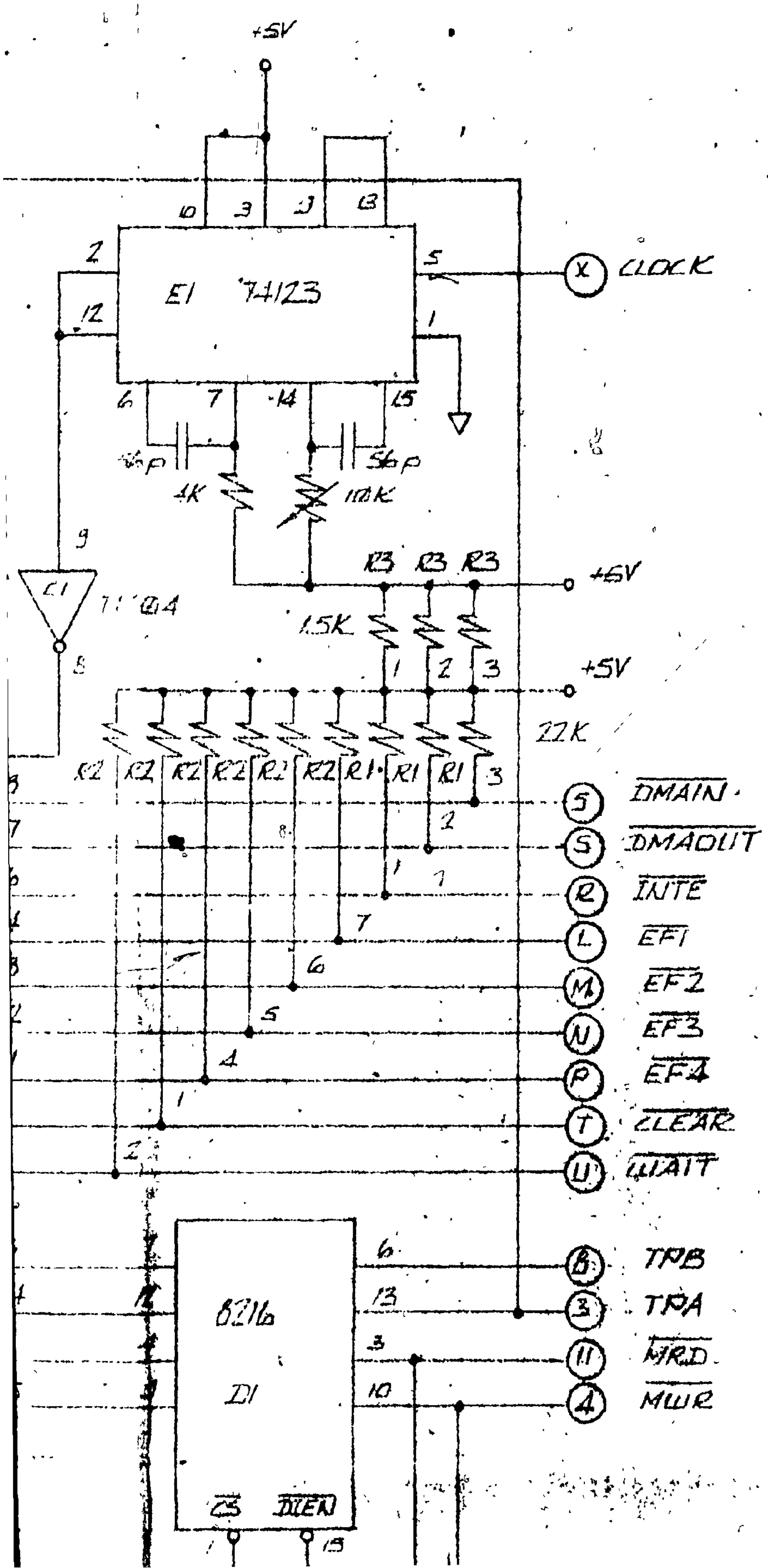
8 M 8


1
2
3
4
5
6
7
8
9
7414
IDEN
W
6
E4
5
MEMDIS
Y
1
E4
2
7414
R2
1.5K
+5V
A7
K
A6
J
A5
H
A4
F
A3
E
A2
D
A1
C
A0
B
15
14
13
12
146


8205
A1
E1
E2
E3
A0
A1
A2
+5V
22K
J1
4042
B1
POL
+5V
8216
D4
D0
D1
D2
D3
DIEN
CS
8216
C3
D4
D5
D6
D7
DIEN
CS
24


INCHES
7 8 9 10
19 20 21 22 23 24 25 26 27
+5V
E1 74123
10 3 9 13 2 12 6 7 14 15
56pF 56pF
4K 100K
R3 1.5K
A7 A6 A5 A4 A3 A2 A1 A0
8216 D2
8216 D3
CS DIEN
C1 74204
+5V
RCA CDP 1802
C2
A7 A6 A5 A4 VDD VCC VSS A3 A2 A1 A0
D0 D1 D2 D3 D4 D5 D6 D7
CLOCK
TPB TPA MRD MWR
SC1 SC0 Q N2 N1 N0
R2 R2 R2 R2 R2 R2 R1 R1
8216 D1
CS DIEN
3 8

26 27 28 29 30 31 32 33 34

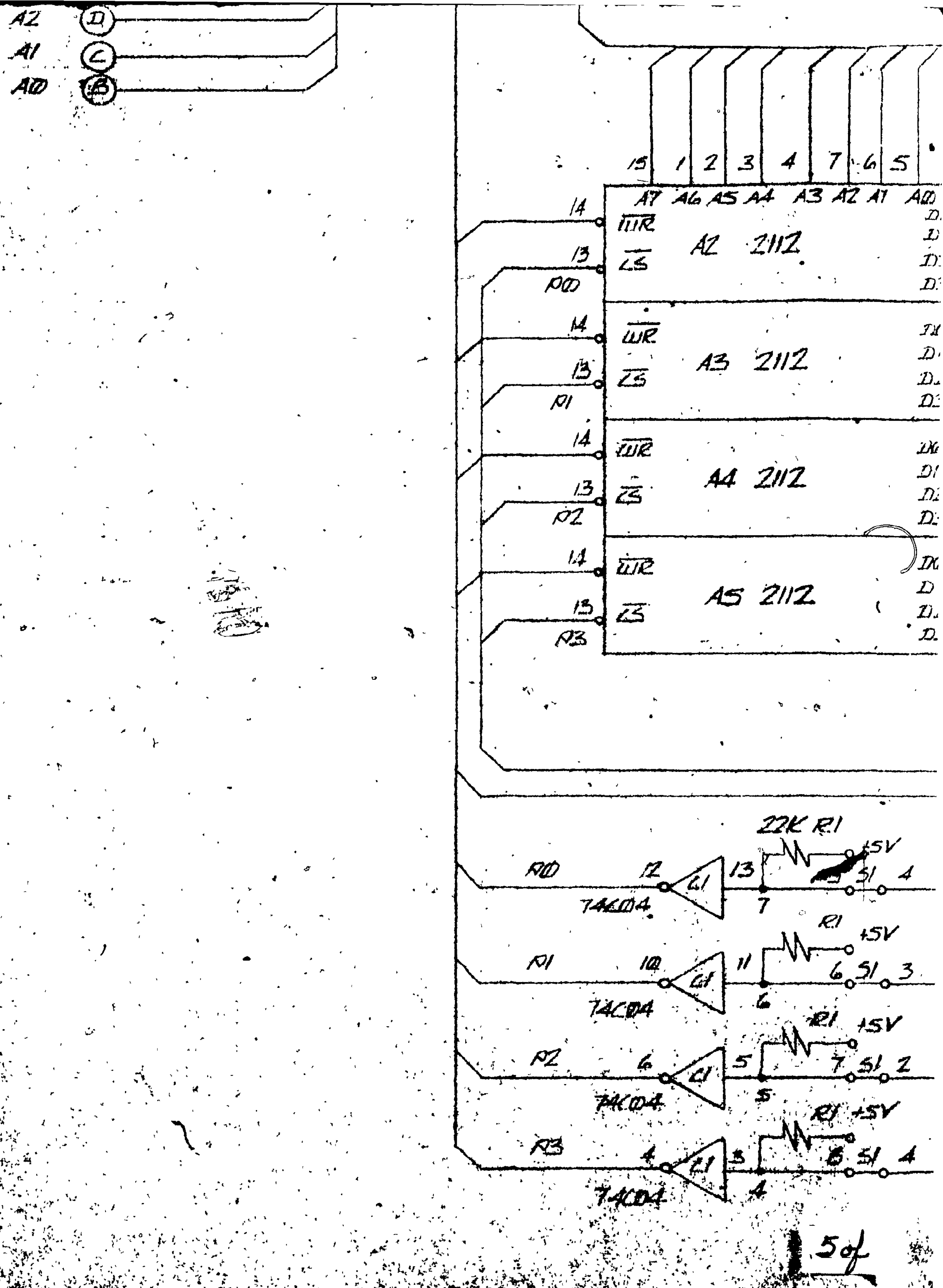


Third Angle Projection
Dimensions In Inches ☐ Mms. ☐
Scale
Tolerances
Material
Finish
Revisions

Iss	FRO DCO	App.	Rel.	Mic.

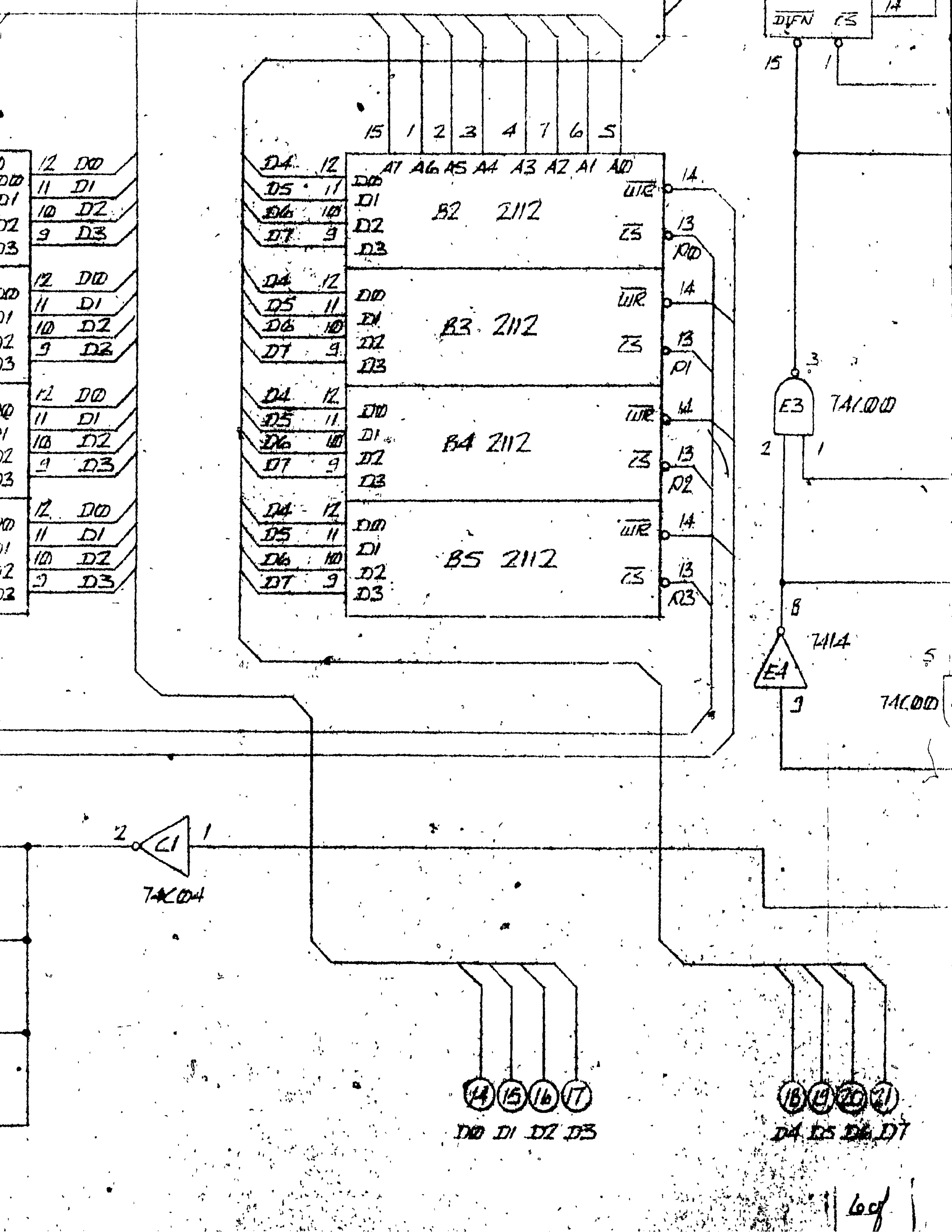
4 of

A B C D E F G H I J K L



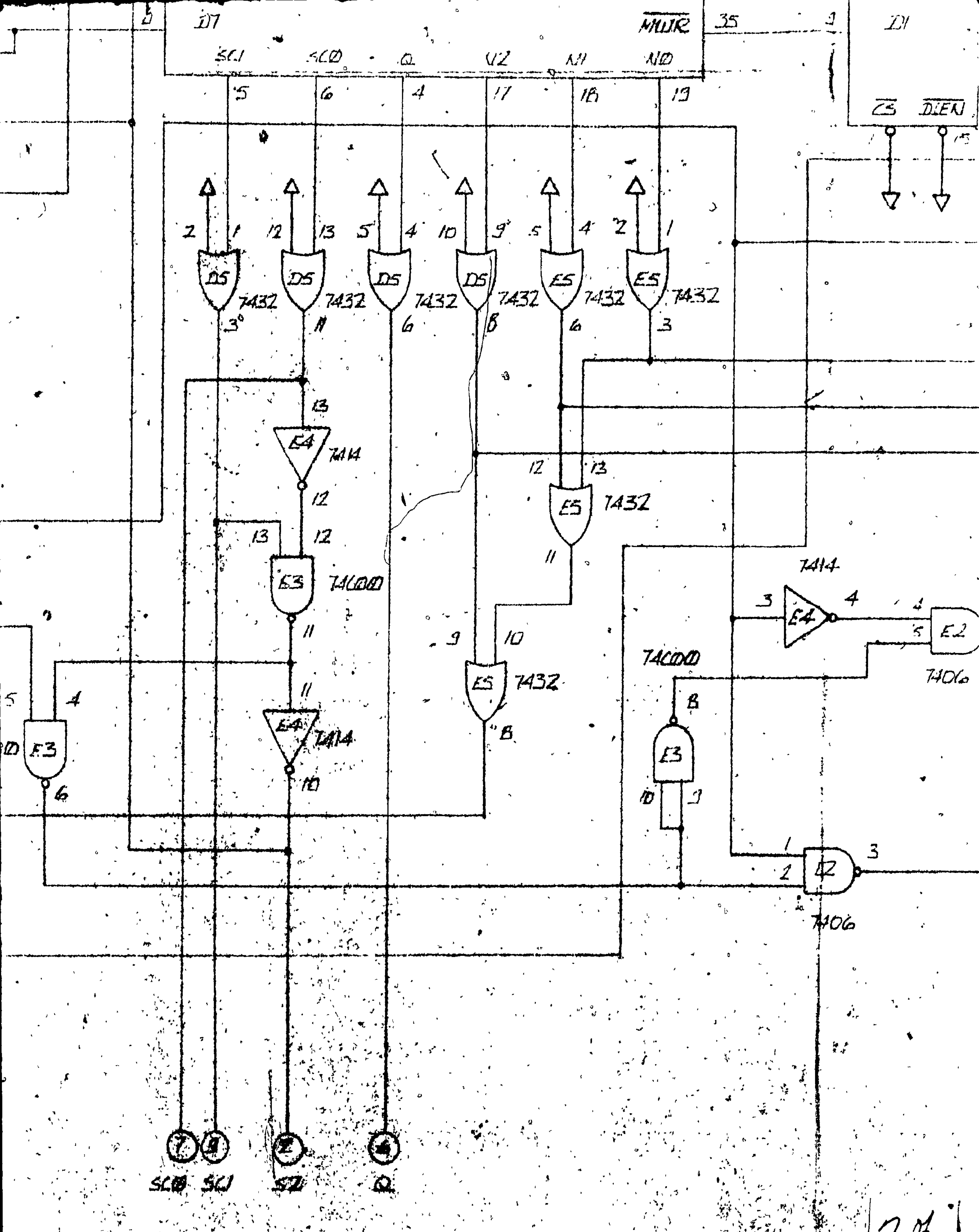
A2
A1
A0
D
C
B
15 1 2 3 4 7 6 5
A7 A6 A5 A4 A3 A2 A1 A0
14
WR
13
CS
R0
A2 2112
A3 2112
R1
A4 2112
R2
A5 2112
R3
22K R1
+5V
R0
12
C1
13
S1
4
7
74C04
R1
10
11
6
3
R2
5
2
R3

5 of

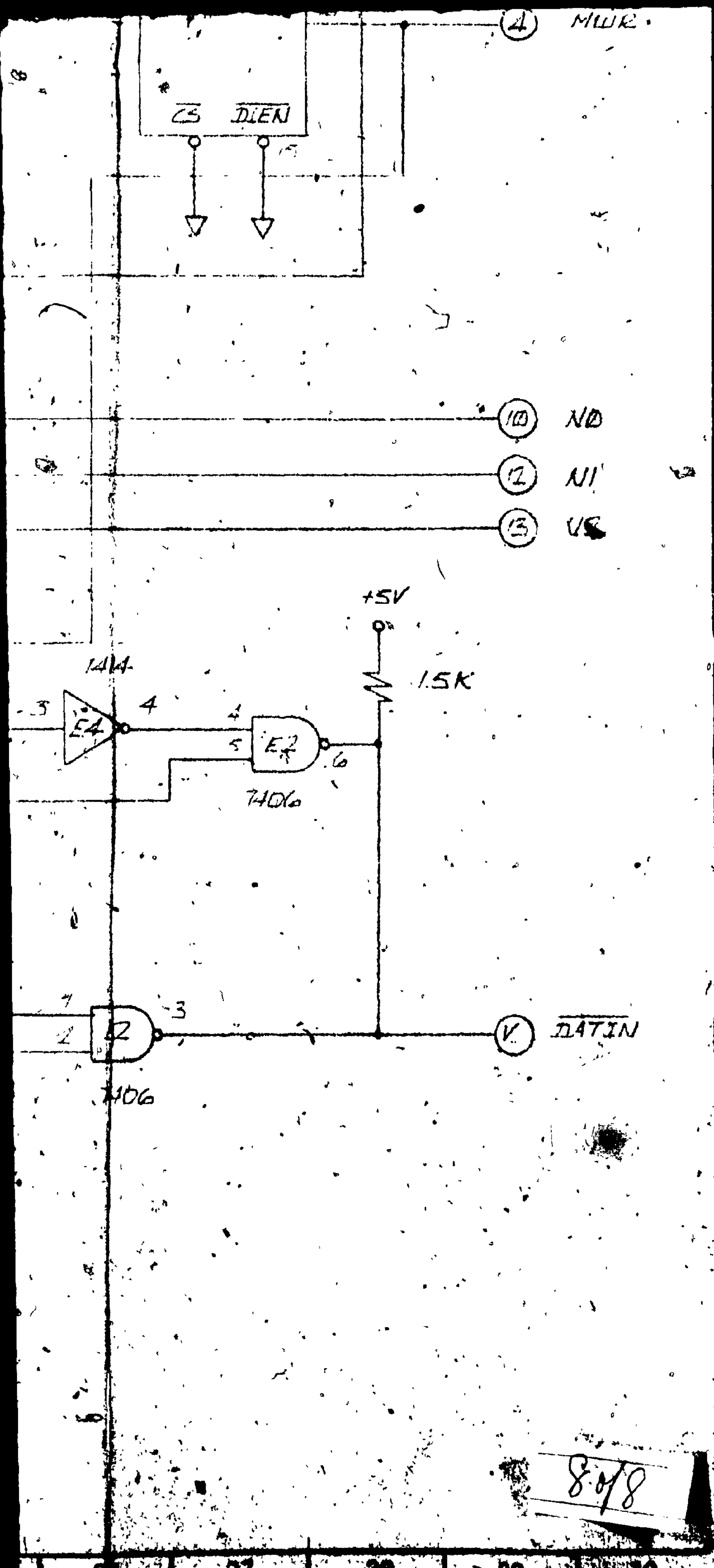


B2 2112
B3 2112
B4 2112
B5 2112
A7 A6 A5 A4 A3 A2 A1 A0
15 1 2 3 4 7 6 5
D0 D1 D2 D3
D4 D5 D6 D7
WR
CS
R0 R1 R2 R3
E3 74C00
E4 7414
74C00
C1 74C04
DIFN CS
14 15 16 17
18 19 20 21
D0 D1 D2 D3
D4 D5 D6 D7

6 of



D7
MWR
35
D1
SC1
SC0
Q
V2
N1
N0
5
6
4
17
18
19
CS
DIEN
D5
E5
7432
E4
7414
E3
74C00
E2
7406
SC0
SC1
S2
Q



Designed	Drawn	Checked
Approved [illegible]	Approved	Approved

Drafting App.	Release APRIL 1977

Reference

Next Ass'y or Family Tree

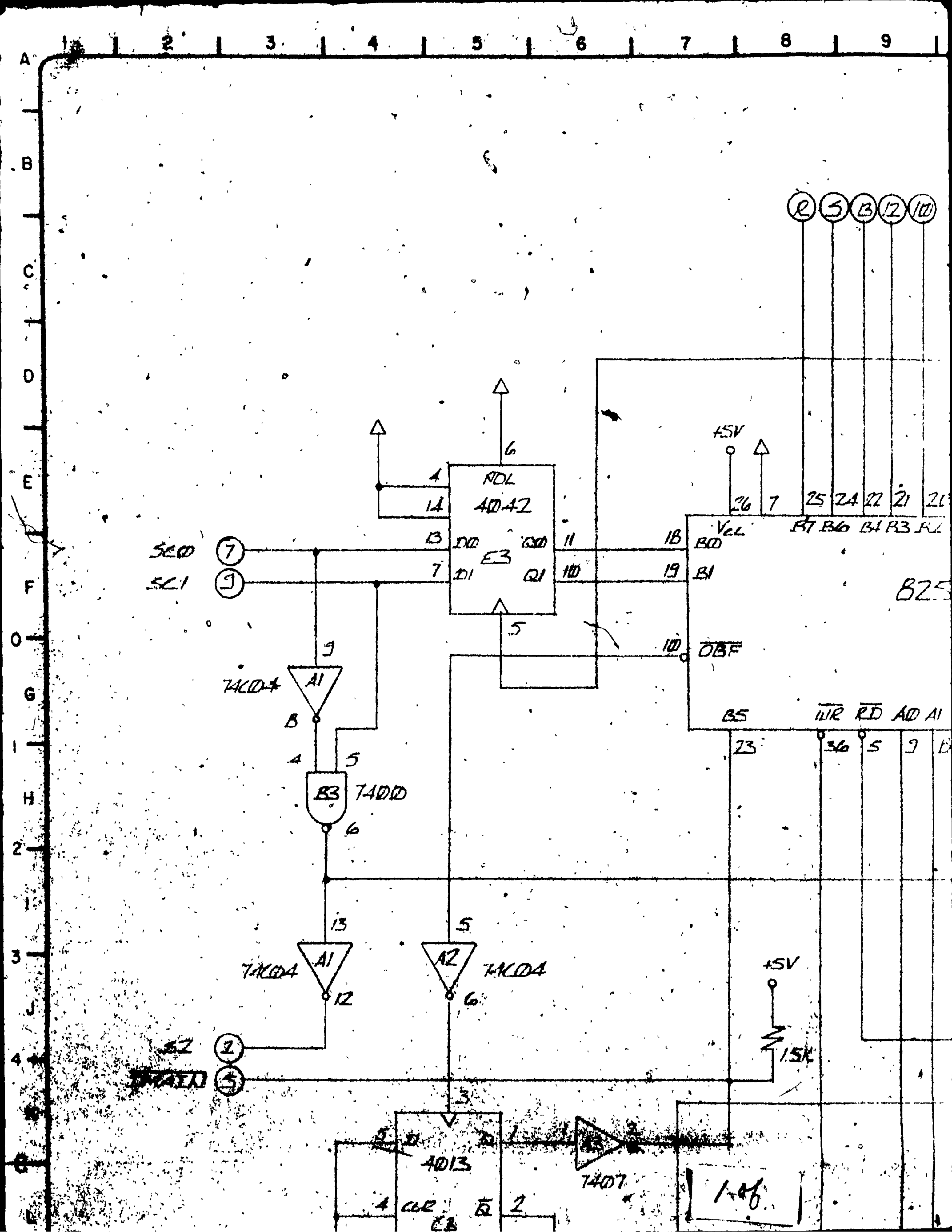
AES

AES Data Ltée/Ltd.

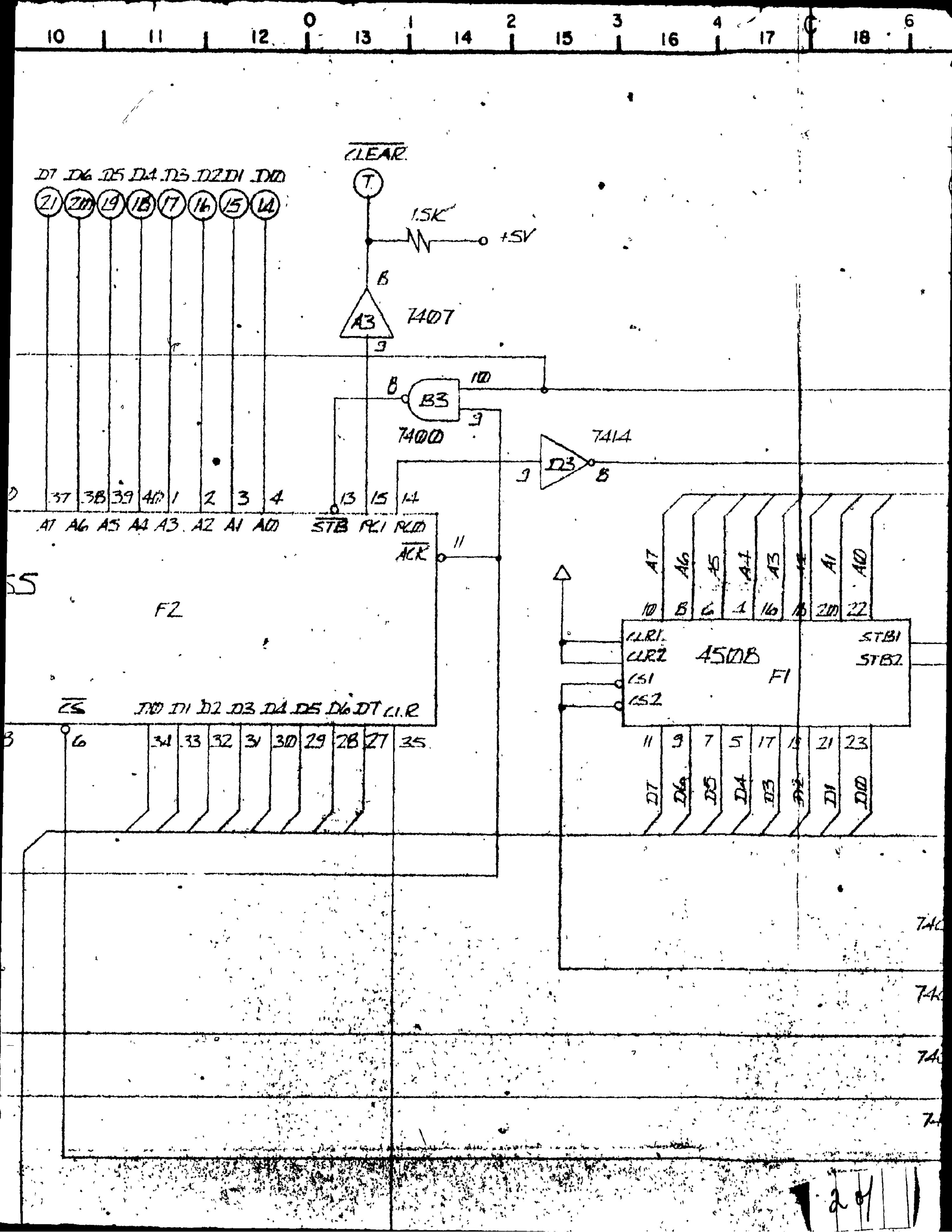
Title CPU

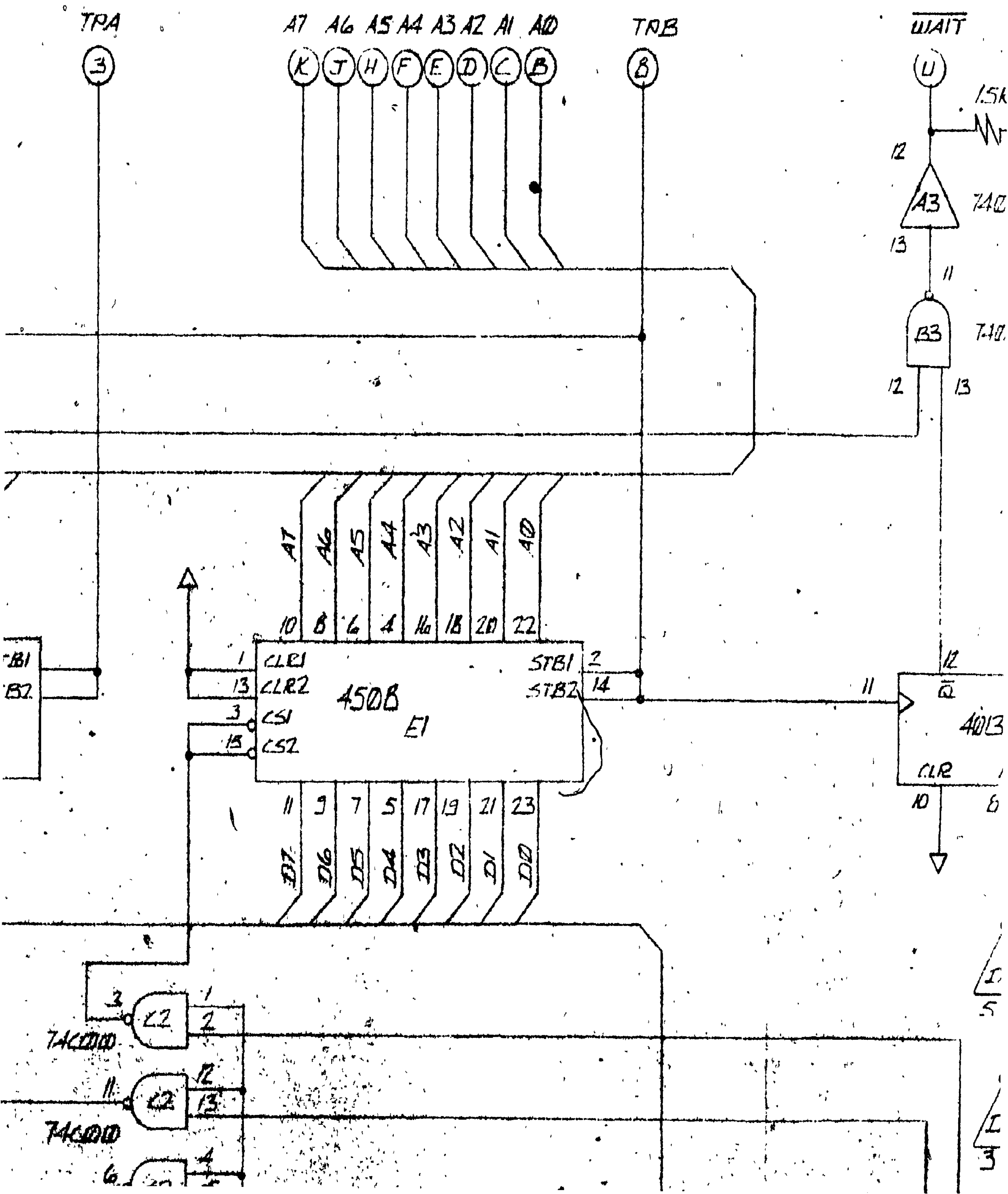
Size	Drawing No.
D	

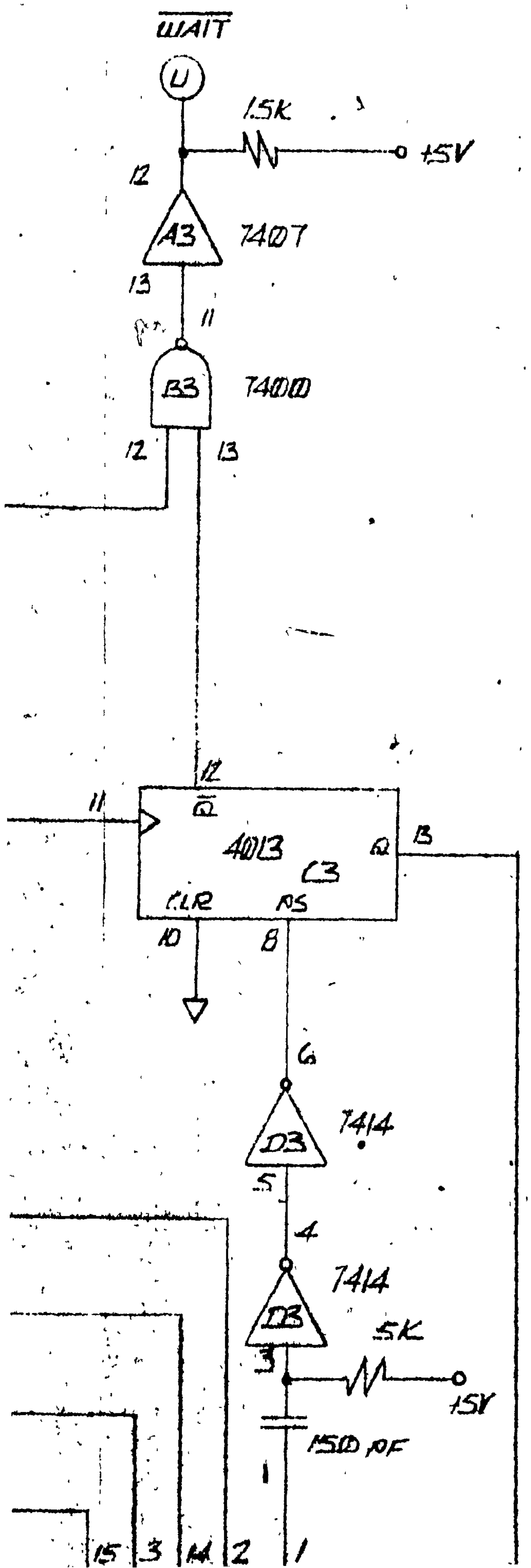
Issue 01	Sheet 2 of 3 Sheets



1
2
3
4
5
6
7
8
9
A
B
C
D
E
F
G
H
J
4042
RDL
E3
D0
D1
Q0
Q1
SC0
SC1
7
9
13
11
10
18
19
14
4
6
5
+5V
26
25
24
22
21
VCC
B0
B1
B7
B6
B4
B3
OBF
825
B5
WR
RD
A0
A1
23
36
74C04
A1
A2
74C00
B3
12
15K
4013
D
Q
CLR
C3
7407
1uf

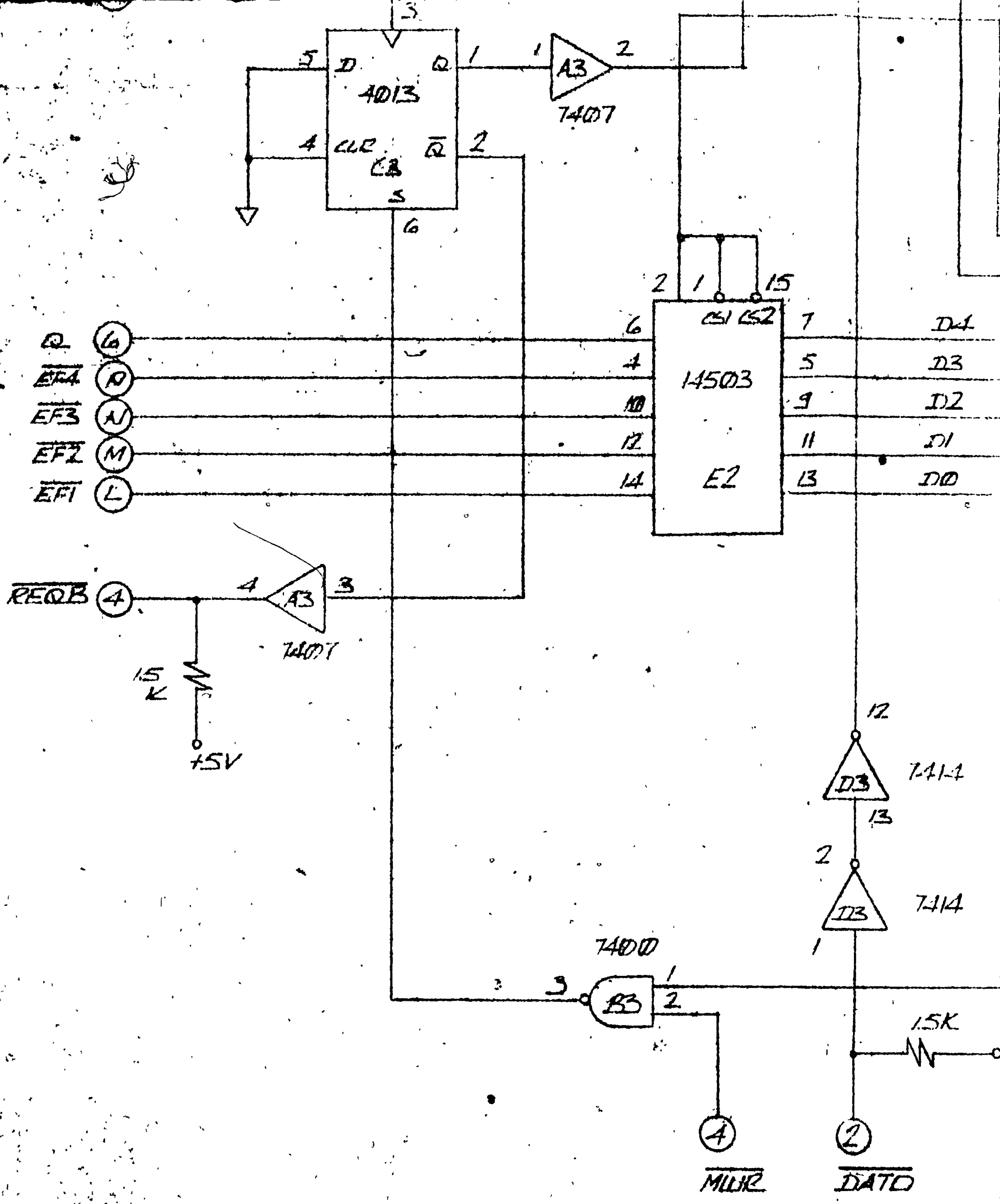

10 11 12 13 14 15 16 17 18
0 1 2 3 4 6
CLEAR
T
1.5K
+5V
7407
A3
B3
7400
10
7414
D3
D7 D6 D5 D4 D3 D2 D1 D0
21 20 19 18 17 16 15 14
37 38 39 40 1 2 3 4
A7 A6 A5 A4 A3 A2 A1 A0
STB PC1 PC0
13 15 14
ACK
11
F2
CS
6
D0 D1 D2 D3 D4 D5 D6 D7 CLR
34 33 32 31 30 29 28 27 35
A7 A6 A5 A4 A3 A2 A1 A0
10 8 6 4 16 18 20 22
4508
F1
CLR1
CLR2
CS1
CS2
STB1
STB2
11 9 7 5 17 19 21 23
D7 D6 D5 D4 D3 D2 D1 D0
74
24


TPA
3
A7 A6 A5 A4 A3 A2 A1 A0
K J H F E D C B
TPB
B
WAIT
U
1.5K
A3
74C
B3
74C
450B
E1
CLR1
CLR2
CS1
CS2
STB1
STB2
D7 D6 D5 D4 D3 D2 D1 D0
4013
Q
CLR
C2
74C00



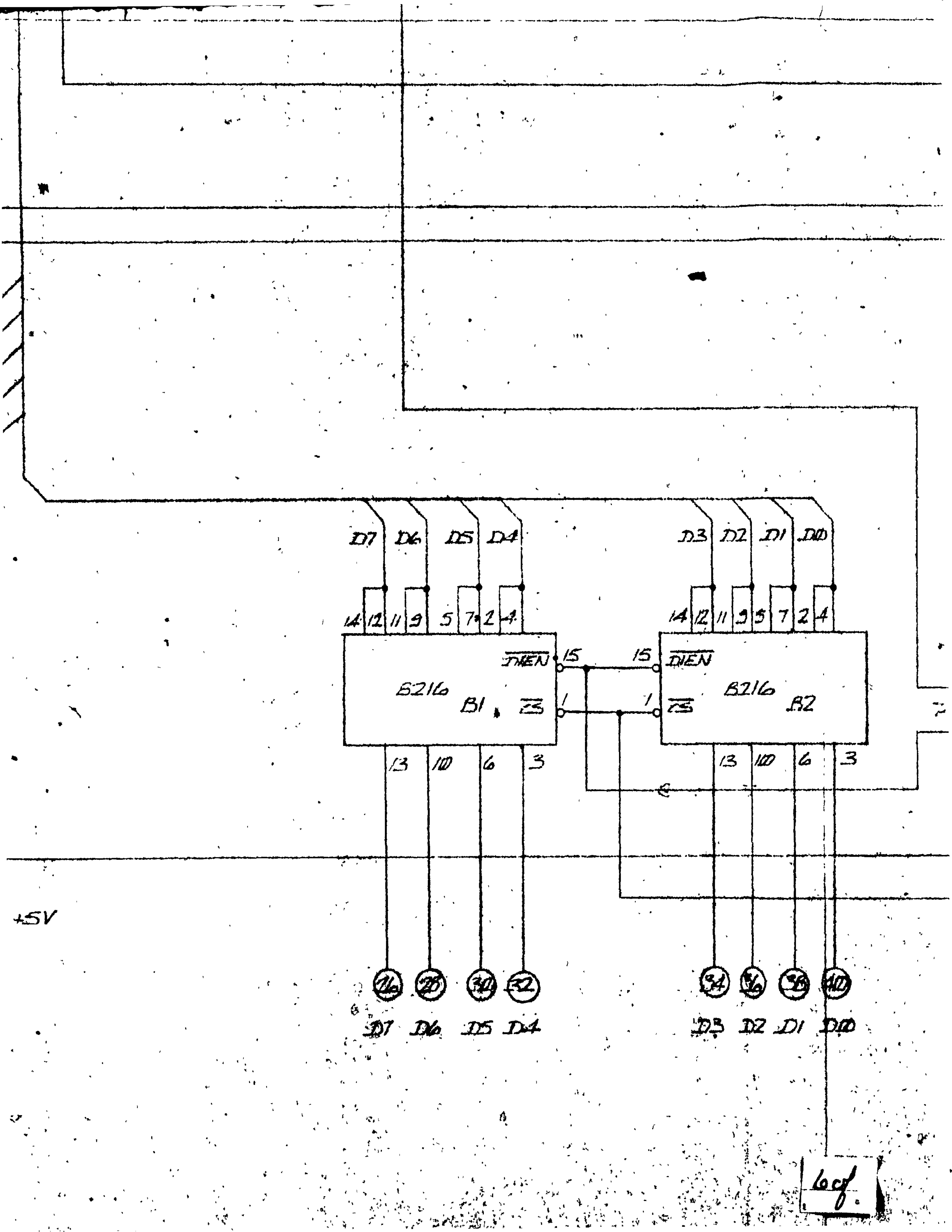
Third Angle Projection	
Dimensions In Inches ☐ Mms. ☐	
Scale	
Tolerances	
Material	
Finish	
Revisions	

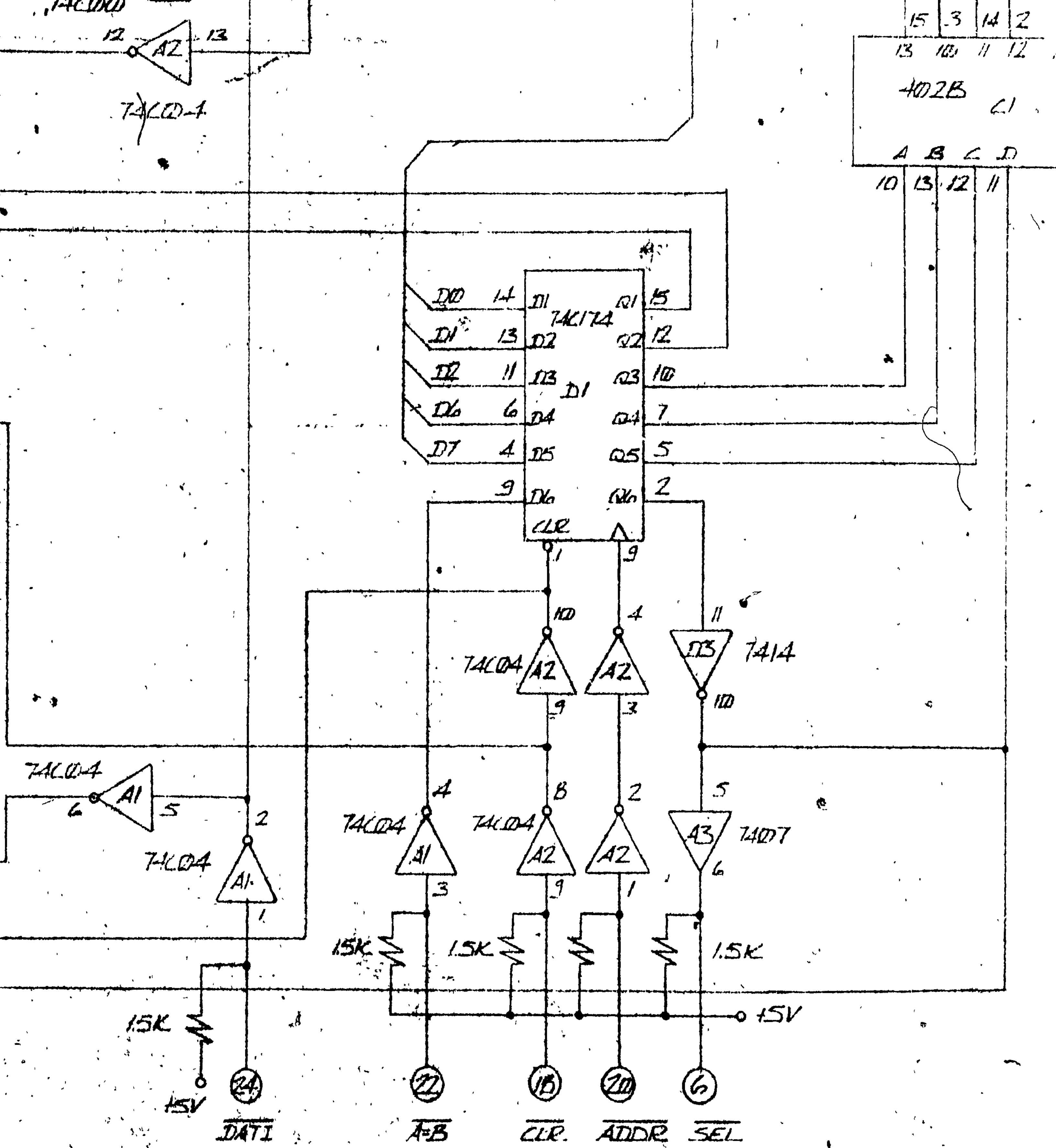
Iss	FRO DCO	App.	Rel.	Mic.



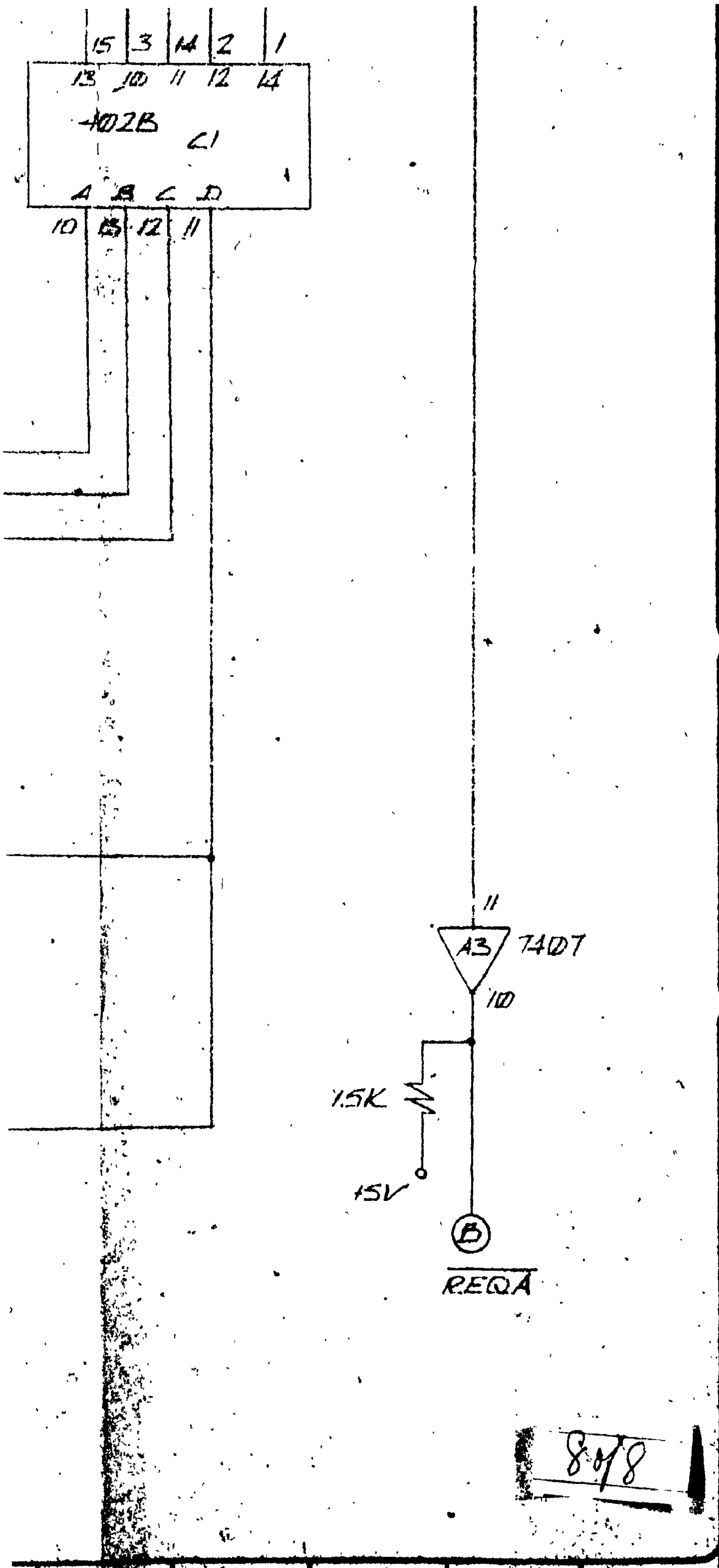
4013
C3
7407
A3
14503
E2
CS1
CS2
Q
EF4
EF3
EF2
EF1
D4
D3
D2
D1
D0
REQB
1.5 K
+5V
7414
D3
7400
B3
1.5K
MWR
DATO

5 of


D7
D6
D5
D4
D3
D2
D1
D0
14 12 11 9 5 7 2 4
8216
B1
DIEN
CS
15
1
8216
B2
13 10 6 3
+5V
26
28
30
32
34
36
38
40
6 of


74C00
A2
74C0-1
402B
C1
74C174
D1
D0
D1
D2
D6
D7
CLR
74C04
A2
7414
D3
74C04
A1
74C04
A1
74C04
A2
A2
A3
7407
1.5K
1.5K
1.5K
1.5K
+5V
+5V
24
22
18
20
6
DATI
A=B
CLR.
ADDR
SEL

7 of



8 of 8

Designed	Drawn	Checked
Approved [illegible]	Approved	Approved

Drafting App.	Release AUG. 1977

Reference

Next Ass'y or Family Tree

AES

AES Data Ltée/Ltd.

Title

MONITOR

Size	Drawing No.
D	

Issue 00	Sheet 8 of 8 Sheets