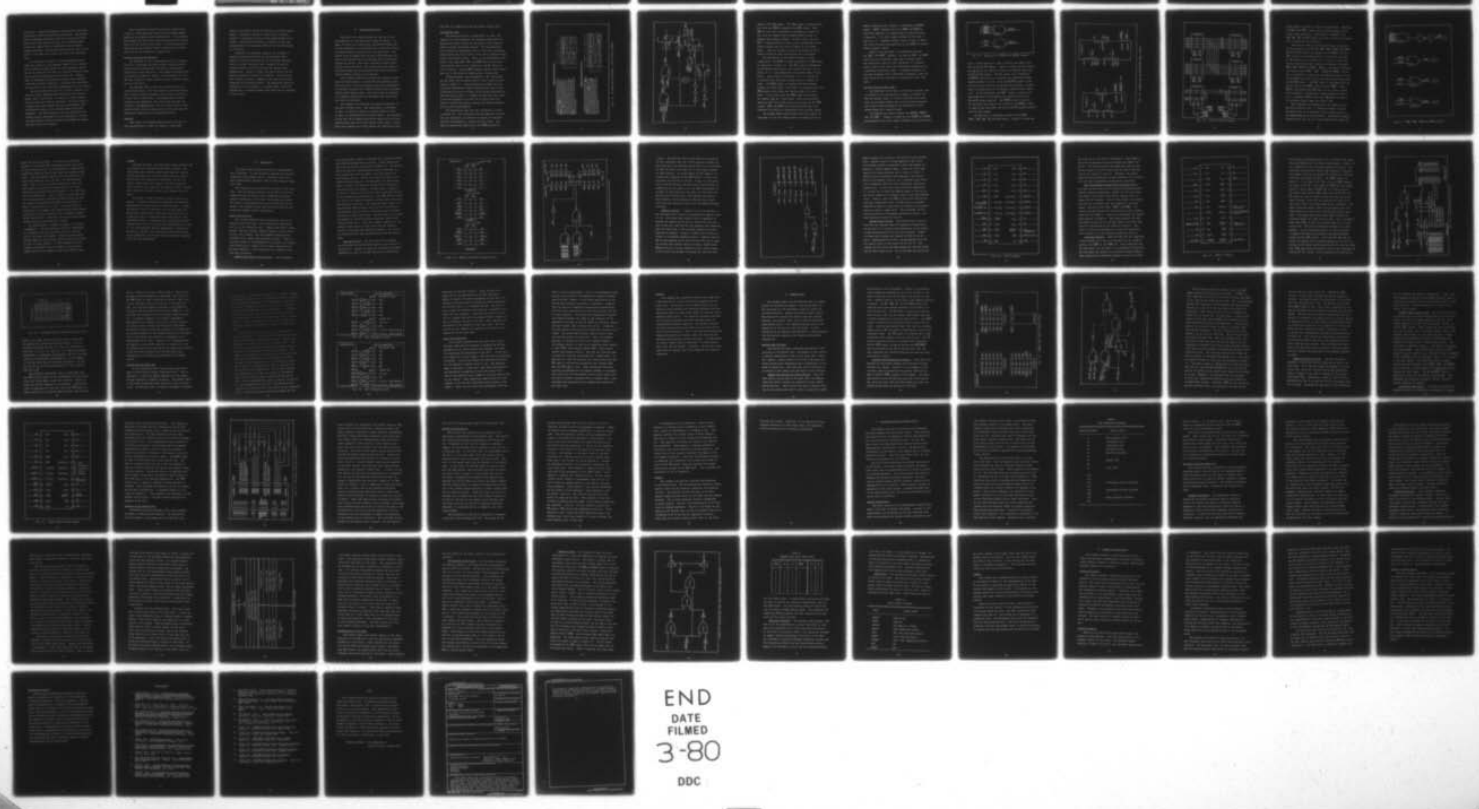
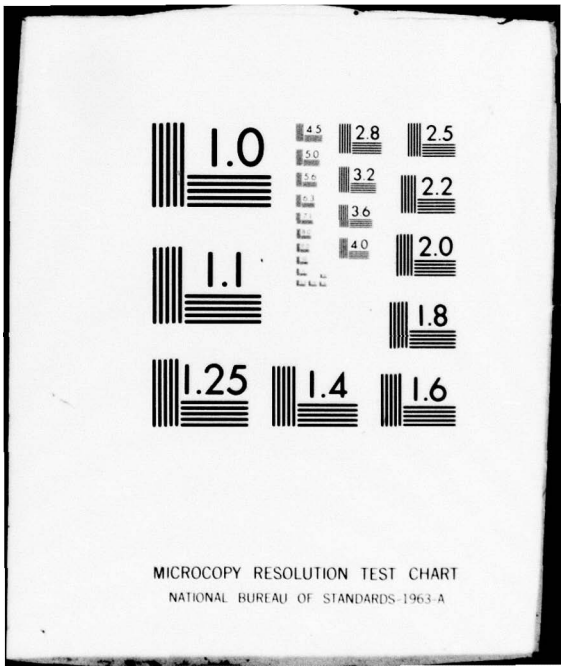
AD-A080 173 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOO--ETC F/G 17/2
PROTOTYPE UNIVERSAL NETWORK INTERFACE DEVICE.(U)
DEC 79 E F BROWN
UNCLASSIFIED AFIT/GE/EE/79-8 NL

| OF |
AD A080173





END
DATE FILMED
3-80
DDC


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
5.6
6.3
7.1
8.0
9.0
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

AD A080173

LEVEL II



DDC FILE COPY

DDC
RECEIVED
FEB 5 1980
A

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY (ATC)

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

DISTRIBUTION STATEMENT A
Approved for public release;
Distribution Unlimited

80 2 5 237

AFIT/GE/EE/79-8

PROTOTYPE UNIVERSAL NETWORK INTERFACE DEVICE

THESIS

AFIT/GE/EE/79-8 Eric F. Brown
Capt USAF

DDC
RECEIVED
FEB 5 1980
A

Approved for public release; distribution unlimited

(14) AFIT/GE/EE/79-8

(6) PROTOTYPE UNIVERSAL NETWORK INTERFACE DEVICE.

(9) Master's THESIS

Presented to the Faculty of the School of Engineering

of the Air Force Institute of Technology

Air Training Command

in Partial Fulfillment of the

Requirements for the Degree of

Master of Science

Accession For
NTIS GRA&I
DDC TAB
Unannounced
Justification
By
Distribution/
Availability Codes
Dist | Avail and/or special
A

(10) by Eric Frederick Brown, B.S.E.E.
Capt USAF
Graduate Electrical Engineering

(12) 88 (11) December 1979

Approved for public release; distribution unlimited

012 225 LB

Acknowledgements

I would like to take this opportunity to express my gratitude to some of those individuals who assisted me in completing this effort. I would like to thank my thesis advisor, Dr. Gary Lamont for his valuable comments on the thesis and for the very interesting learning experience he allowed me to undertake. I would also like to thank my readers for their encouragement and aid in this study. The technicians in the lab provided a high quality of support, which saved me much time. I have to thank Dan Zambon, in particular, for the excellent job of wire-wrapping he did on the device. Lastly, I have to thank my wife, Kathy, for her immeasurable aid in helping me to complete my thesis. I do not think I could have finished the thesis without her.

Contents

	Page
Acknowledgements	ii
List of Figures	v
List of Tables	vi
Abstract	vii
I. Introduction	1
Background	1
Basic Design Approach	3
Problem Statement and Approach	5
Overview	5
II. Dual Processor Card	7
Arbitration Logic	8
Bus and Control Signal Logic	12
Summary	19
III. Local Card	20
Local Card Circuitry	20
Address and Control Signal Drivers	20
Data Bus Drivers	20
Address Decoding	24
Counter Timer Circuit	26
2651 Programmable Communications Interface (PCI)	28
Interrupt Handling	28
Features of the Local Card	33
Local Card Operations	36
Summary	38
IV. Network Card	39
Network Card Circuitry	39
Address Bus and Control Signal Drivers	39
Data Bus and Controlling Circuitry	40
Address Decoding Circuitry	44
Counter Timer Circuit Signal	45
Z-80 Serial Input/Output	45

Page

Features of the Network Card 47
Network Card Operations 50
Summary 52

V. Universal Network Interface Device 54

Overall Architecture 54
Procedures Required Before Use 57

Hardware Procedures 57
Software Procedures 59

Implementation of the UNID 63

Out-of-Phase Clock Signal 64
Addressing PROM 65
Mini-Disk Interface 67
UNID Wiring 68

Summary 69

VI. Summary and Conclusions 70

Overview of Thesis 70
Follow-On Efforts 70
Changes in Device Design 73
Recommended Projects 74

Bibliography 75

Vita 77

List of Figures

Figure		Page
2-1	OP Code Fetch and Memory Access Timing Diagrams	9
2-2	Arbitration Logic	10
2-3	Generation of $\overline{\text{STROBE}}$ and $\overline{\text{STROBE}}'$ Signals	13
2-4	Address Bus and $\overline{\text{RD}}$ and $\overline{\text{WR}}$ Signal Control	14
2-5	Data Bus Control Signals	15
2-6	$\overline{\text{MREQ}}$, $\overline{\text{RFSH}}$, $\overline{\text{XWAIT}}$ and $\overline{\text{YWAIT}}$ Signals	17
3-1	Address and Control Signal Drivers	22
3-2	Data Bus Drivers and Controlling Circuitry	23
3-3	Address Decoding	25
3-4	CTC #1 Signals	27
3-5	USART #1 Signals	29
3-6	Interrupt Handling Circuitry	31
3-7	Interrupt Request/Acknowledge Cycle	32
3-8	Port Configured as DTE	35
3-9	Port Configured as DCE	35
4-1	Address and Control Signal Drivers	41
4-2	Data Bus Controlling Circuitry	42
4-3	Counter Timer Circuit Signals	46
4-4	SIO Signals and Port Configuration	48
5-1	Typical Load Map	62
5-2	Combinational Logic Used to Replace Addressing PROM	66

List of Tables

Table		Page
I	UNID Connector Interfaces	56
II	Combinational Logic Truth Table	67
III	Color Coding of Wiring	68

Abstract

The thesis describes a prototype version of a flexible message processor designed for computer communications network applications. This message processor is microcomputer based and is called a Universal Network Interface Device. The thesis describes the operational features and construction details of the three cards, which were developed in the construction of the prototype device. These three cards are used with two microcomputer boards and a memory board to provide the capability to interface local users to a computer communications network. The device also provides the capability to act as the node connecting two networks operating under dissimilar protocols.

PROTOTYPE UNIVERSAL NETWORK INTERFACE DEVICE

I. Introduction

The purpose of this investigation was to perform an implementation of a microcomputer based message processor with the inherent flexibility which would allow it to be used in different types of data communications network applications. Using a preliminary design, the message processor was implemented using a modular approach. Since the device is microcomputer based, the resident software can be modified to configure the device for the required type of message processor of a given application.

The remaining sections of this chapter will address background information relative to this effort, the basic design of this message processor which is called a Universal Network Interface Device (UNID), the problem statement and approach, and an overview of the subjects covered in this report.

Background

The 1842 Electronics Engineering Group (EEG) completed a report on 31 OCT 77 entitled An Engineering Assessment Toward Economic, Feasible and Responsive Base-Level Communications Through the 1980's (Ref 1). This report was written to provide an engineering input to the Air Force Communications Service (AFCS) planning process for meeting base-level

telecommunications needs. The report reviews current and planned base telecommuncations requirements and capabilities. It also provides a technological forecast and the possible effect of technology on user requirements. The last part of the report describes the composite scenario of "typical" base-level telecommunications requirements and capabilities through 1990. The last two parts of the 1842 EEG report introduced the concept of a multi-ring network as being the prime candidate for a base-level telecommunications network. This multi-ring network offered particular advantages in terms of cost, evolutionary implementation and flexibility. A key to the multi-ring concept was the development of five different devices which could interface the multiple rings together.

Rome Air Development Center (RADC) was tasked with addressing the problem of the interface devices. It was recognized that the various interface devices proposed had similar features and were required to perform similar functions. Thus it appeared reasonable that a single, albeit flexible interface device could be developed which could meet the separate requirements of each of the five proposed interface devices of the multi-ring concept. This idea was incorporated into a postdoctoral study program, and later became the basis of a preliminary research investigation (Ref 12). The report of the investigation, entitled Preliminary Design of a Universal Network Interface Device, was the first phase of an effort to develop the required

message processor. The investigation involved definition of the functional requirements of the device, translation of these requirements into an overall system design, design of the hardware of the device, and the development of the software routines to verify the correct operation of the device. The referenced document was the principal source of information for the actual implementation. The overall design of the device as proposed was excellent and it was this general design which was employed in implementing the device.

Basic Design Approach

The basic design of the device involved two Z-80 Microcomputer Boards which shared a common block of memory. Each of the microcomputer boards will perform specific tasks within the framework of the device. To meet the "universal" qualification the UNID would have to interface to users directly or through Modems. Also, the device must be able to interface with a data communications network. To be able to perform these two types of interfacing two special cards were designed, the Local Card and the Network Card.

The design as specified in the previous effort was chosen to provide modularity of the hardware so that the device could be increased in capability by the addition of one or more Local or Network Cards. The Local Card provided the means to interface up to four RS-232C users with

the device. Through the addition of more Local Cards additional users could be serviced by the device. The Network Card was designed around the Z-80A SIO, since the SIO was part of the Z-80A family of chips and the SIO provided significant capability for interfacing with different network protocols. Two Network Cards would be used in the device if the UNID was to act as an inter-ring interface mode.

As previously mentioned, Z-80 Microcomputer Boards (MCB) were chosen as the processor boards in this design. The preliminary design called for Z-80A processor boards due to the speed of the Z-80A, its vectored interrupt capability and its sophisticated instruction set. However, when the hardware was procured no commercially available Z-80A MCB's existed; therefore, Z-80 MCB's were purchased and were used in the prototype implementation in this effort.

The basic design did call for a block of memory to be shared to allow communications between the two microcomputers used in the device. To meet this design feature, the Dual Processor Card was designed. This Card acted as a memory arbitrator between the two microcomputers when simultaneous accesses to the shared memory block were attempted. The Dual Processor Card also arbitrated the refresh signals developed by the two processors, so that the shared memory would be correctly refreshed and memory accesses would not interfere.

Thus the overall design of the device would be described as a dual processor device with one shared memory partition to allow processor communication. One processor would handle the data flow to and from local users through the use of one or more Local Cards. The second processor would process the network traffic through use of the Network Card and the processor's capabilities.

Problem Statement and Approach

The objective of this investigation was the implementation of a prototype Universal Network Interface Device. This objective included both the hardware and software required for device operation. The hardware implementation is essentially complete; however, the development of software was limited to simple routines used in testing various parts of the device.

The approach taken to meet the stated objective involved three phases. The first phase of the investigation was a study of previous research and of material relating to final uses of the device. The second phase involved testing of the individual cards after they had been constructed in breadboard form. The last phase of this investigation dealt with the construction of the cards and the associated integration of the prototype device.

Overview

Much effort was expended during certain portions of this investigation in order to attempt to understand

previous work and to verify the operation of boards implemented. Discussion related to these areas is limited because it contributes little to the understanding of the effort. Instead the discussion is centered on what was accomplished on the prototype UNID and the various aspects of the device.

Therefore the subsequent chapters are intended to provide a thorough description of the prototype UNID. Chapter II provides descriptions of the design and operation of the Dual Processor Card. Chapters III and IV likewise address the Local Card and the Network Card, respectively. Chapter V covers the entire device and the specific features involved in the design. The sixth and final chapter provides a summary of the thesis and recommendations for follow-on effort. The appendices which are referenced to are included in a second report, since the drawings are on oversized paper which could not be included in this report.

II. Dual Processor Card

The key to the overall design was the use of two microcomputers as the processors within the device. In order for data to be effeciently transferred between the two microcomputers it was decided that a particular block of memory should be shared by the two processors. This shared block of memory would contain tables defining the status and characteristics of messages being transferred through the device. Also when a message is being transferred through the device, the message would be stored in the shared memory by one processor which would allow the second processor access to the message.

The circuitry required to manage the requests to the block of shared memory is called the Dual Processor Card. This card uses the control signals from each microcomputer to arbitrate accesses to a shared 16K byte block of dynamic Random Access Memory (RAM). This 16K of RAM is contained on a separate board which was purchased from Zilog along with the two Z-80 MCB's.

This chapter will describe the detailed operation of the Dual Processor Card. This description is provided, so that, in follow-on efforts a minimal amount of time is spent in understanding the current design. The circuitry on the card can be apportioned rather easily into the arbitration logic and the bus and control signal logic. Therefore, this chapter will first examine the arbitration logic

and then will address the bus and control signal logic.

Arbitration Logic

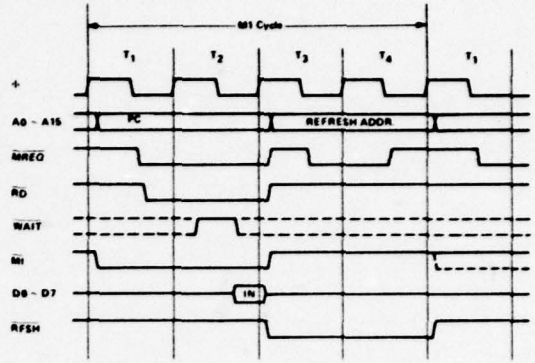
The arbitration logic is duplicated, in that, the signals from each processor are processed identically to develop the proper signals at the proper time to perform memory accesses and memory refresh. The two processors were arbitrarily designated X and Y, where the X processor would process the local traffic while the Y processor would handle the network traffic. Thus, in the descriptions that follow, the terms $\overline{XA15}$, $\overline{YM1}$, and $\overline{MREQ}$ refer to the high-order address line of the X processor, the $\overline{M1}$ signal of the Y processor, and the $\overline{MREQ}$ (memory request) signal which goes out to the block of shared memory, respectively.

Timing diagrams, with explanations, for instruction op code fetch and for memory read or write cycles are provided in Figure 2-1. These diagrams are taken from the Zilog Z80-CPU/Z80A-CPU Product Specification (Ref 17) and illustrate the relative timing of the signals processed by the arbitration logic. It should be noted that the clocks of the microcomputers are 180° out of phase with each other to insure that memory accesses and memory refreshes do not occur simultaneously.

The arbitration logic for the X processor is shown in Figure 2-2. The Y processor uses an identical circuit; the only difference, is that the X signals are replaced with the corresponding Y signals and vice versa. The XA14 is inverted and AND'ed with the $\overline{XRFSH}$ and XA15 to

INSTRUCTION OP CODE FETCH

The program counter content (PC) is placed on the address bus immediately at the start of the cycle. One half clock time later $\overline{MREQ}$ goes active. The falling edge of $\overline{MREQ}$ can be used directly as a chip enable to dynamic memories. $\overline{RD}$ when active indicates that the memory data should be enabled onto the CPU data bus. The CPU samples data with the rising edge of the clock state T_3. Clock states T_3 and T_4 of a fetch cycle are used to refresh dynamic memories while the CPU is internally decoding and executing the instruction. The refresh control signal $\overline{RFSH}$ indicates that a refresh read of all dynamic memories should be accomplished.



MEMORY READ OR WRITE CYCLES

Illustrated here is the timing of memory read or write cycles other than an OP code fetch (M_1 cycle). The $\overline{MREQ}$ and $\overline{RD}$ signals are used exactly as in the fetch cycle. In the case of a memory write cycle, the $\overline{MREQ}$ also becomes active when the address bus is stable so that it can be used directly as a chip enable for dynamic memories. The $\overline{WR}$ line is active when data on the data bus is stable so that it can be used directly as a R/W pulse to virtually any type of semiconductor memory.

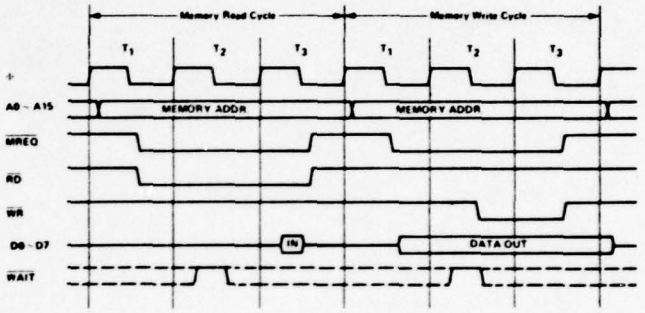


Fig. 2-1. OP Code Fetch and Memory Access Timing Diagrams

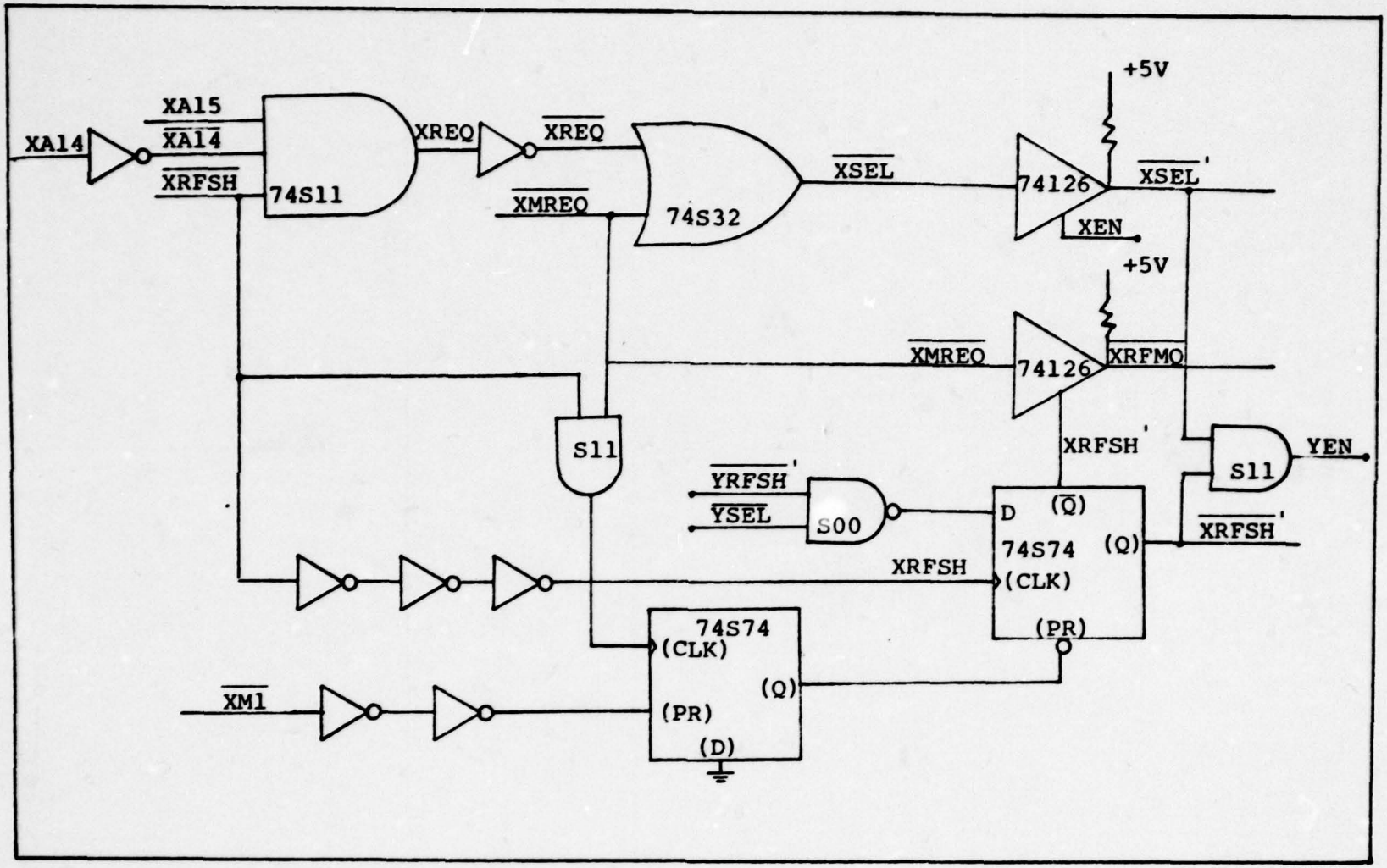


Fig. 2-2. Arbitration Logic

generate the XREQ signal. The XREQ signal is inverted and OR'd with the $\overline{\text{XMREQ}}$ to generate the $\overline{\text{XSEL}}$ signal. When $\overline{\text{XSEL}}$ is true, the X processor is attempting a memory access into the shared block of memory which resides in address space 8000_H to $BFFF_H$. If XEN is high, then the $\overline{\text{XSEL}}'$ is generated and is later used in developing the bus control signals and the control signals to the shared memory. XEN will be high as long as no memory access or memory refresh has been initiated by the Y processor.

The arbitration of the refresh signals is more complicated. The $\overline{\text{XRFSH}}$ is always preceded by an $\overline{\text{XM1}}$ signal as indicated in Figure 2-1. The $\overline{\text{XM1}}$ signal is delayed going through two 74L04 gates, yet causes flip-flop 1 to be preset to "1", which deactivates the preset input of flip-flop 2. The D input of flip-flop 2 is a "0" unless the Y processor has initiated a memory access or memory refresh. The $\overline{\text{XRFSH}}$ signal is inverted and then delayed through two 74L04 gates. This delay is necessary since the $\overline{\text{MREQ}}$ signal occurs after the trailing edge of a clock pulse. On the other hand the $\overline{\text{RFSH}}$ signal occurs after the leading edge of a clock pulse. With the processors operating 180° out of phase and with no delay on the $\overline{\text{RFSH}}$ signals, $\overline{\text{XSEL}}$ and $\overline{\text{YRFSH}}$ were occurring at the same time. Therefore the delay was implemented on the refresh signal.

The delayed XRFSH signal would cause the Q output of flip-flop 2 to go low (conditioned on no memory access or

memory refresh by the Y processor) providing the $\overline{\text{XRFSH}}'$ signal. $\overline{\text{XRFSH}}'$ remained low until $\overline{\text{XMREQ}}$ and $\overline{\text{XRFSH}}$ go high which provides the leading edge to clock flip-flop 1 causing a "0" at the Q output presetting flip-flop 2 back to a "1". Meanwhile XRFSH' signal is used to enable the 74126 and allow the generation of the $\overline{\text{XRFMQ}}$ (X refresh memory request) signal.

Finally, the YEN signal is developed by AND'ing the $\overline{\text{XSEL}}'$ and $\overline{\text{XRFSH}}'$ together. As long as $\overline{\text{XSEL}}'$ and $\overline{\text{XRFSH}}'$ were high the Y processor could make a memory access. The arbitration logic for the Y processor is a perfect mirror image of the logic described above relating to the X processor arbitration logic. A complete schematic of the Dual Processor Card is provided in Appendix A and the dual nature of the arbitrator is reflected in that schematic.

Bus and Control Signal Logic

The remainder of the logic on the Dual Processor Card is involved in the generation of signals to control the data and address buses, to control read and write operations on the shared memory and to place either processor into a WAIT condition pending the availability of the shared memory for a memory access.

The address bus control signals are $\overline{\text{STROBE}}$, $\overline{\text{STROBE}}'$, YEN, and $\overline{\text{XSEL}}'$. Figure 2-3 shows how the $\overline{\text{STROBE}}$ and $\overline{\text{STROBE}}'$ are generated from other signals produced by the arbitration

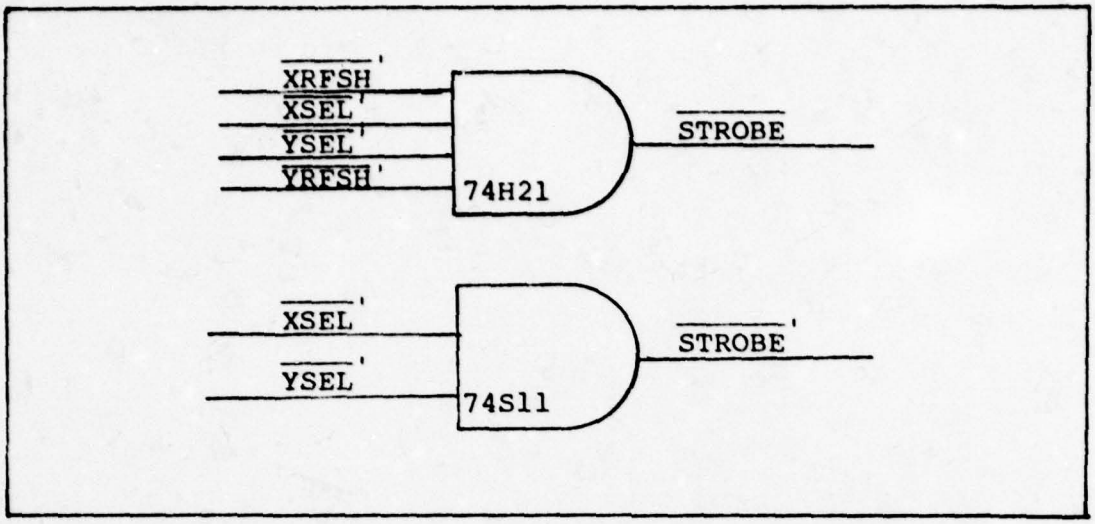


Fig. 2-3. Generation of $\overline{\text{STROBE}}$ and $\overline{\text{STROBE}}'$ Signals

logic. These signals are used to control the address bus and the $\overline{\text{RD}}$ and $\overline{\text{WR}}$ signals. Figure 2-4 shows how the signals are applied to the 74S157s to control the address buses and the $\overline{\text{RD}}$ and $\overline{\text{WR}}$ signals. The YEN signal can be defined as a $\overline{\text{SELECTX}}$ signal since it indicates an X processor memory operation when it is low. Thus the $\overline{\text{SELECTX}}$ signal is used on the low order byte of the address bus selectors, because data is transferred on A7-A0 during a memory refresh operation. On the other hand, the $\overline{\text{XSEL}}'$ signal is used on the high-order byte of the address bus selectors and on the $\overline{\text{RD}}$ and $\overline{\text{WR}}$ signal selector. The $\overline{\text{STROBE}}$ signal is used on the low-order byte address bus selectors and $\overline{\text{STROBE}}'$ signal is used on the high-order byte and on the $\overline{\text{RD}}$ and $\overline{\text{WR}}$ selectors for the same reasons.

The data bus is controlled by means of the $\overline{\text{XSEL}}'$, $\overline{\text{YSEL}}'$, $\overline{\text{XWR}}$, $\overline{\text{YWR}}$, $\overline{\text{XRD}}$ and $\overline{\text{YRD}}$ signals. Figure 2-5 shows how

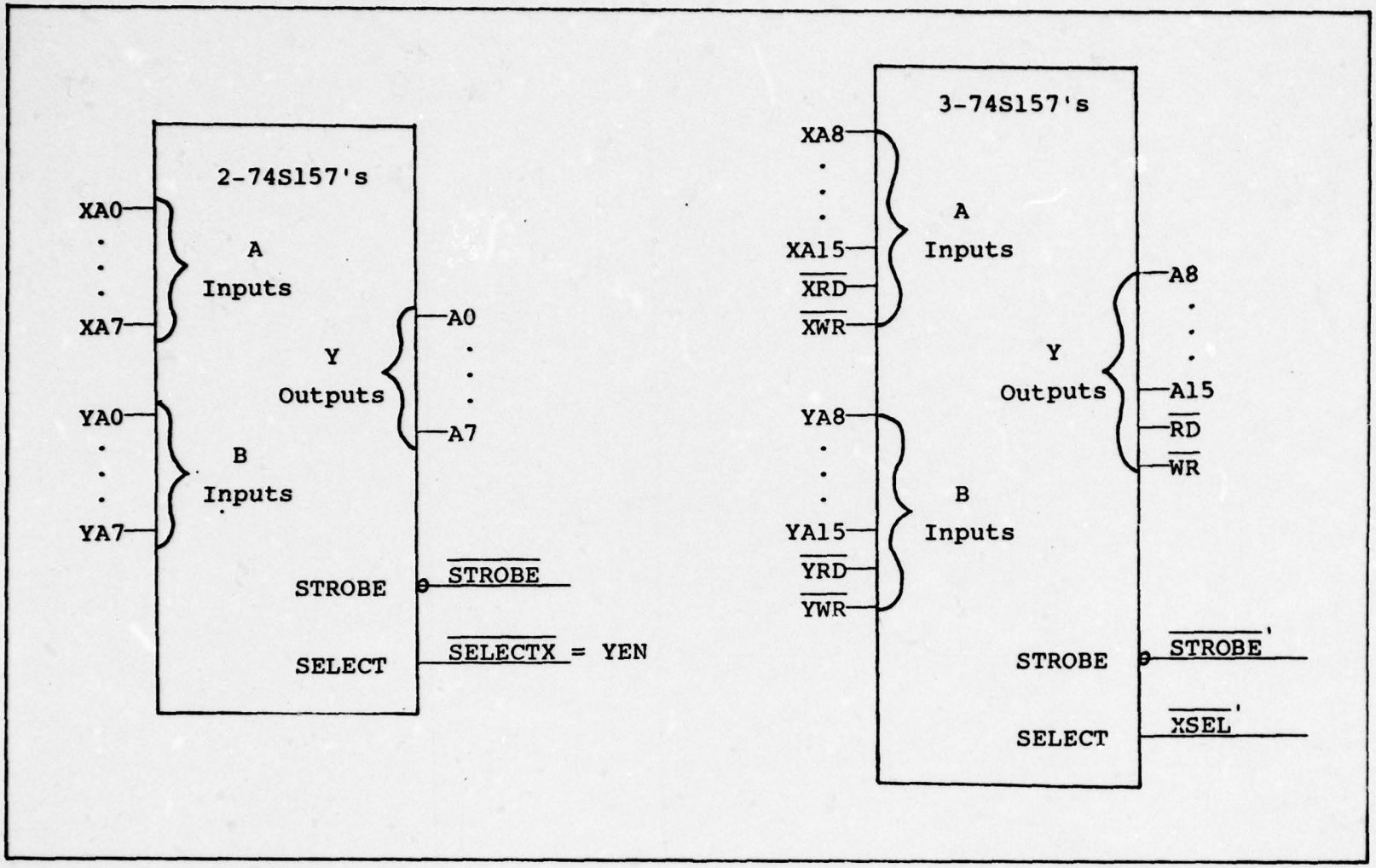


Fig. 2-4. Address Bus and $\overline{RD}$ and $\overline{WR}$ Signal Control

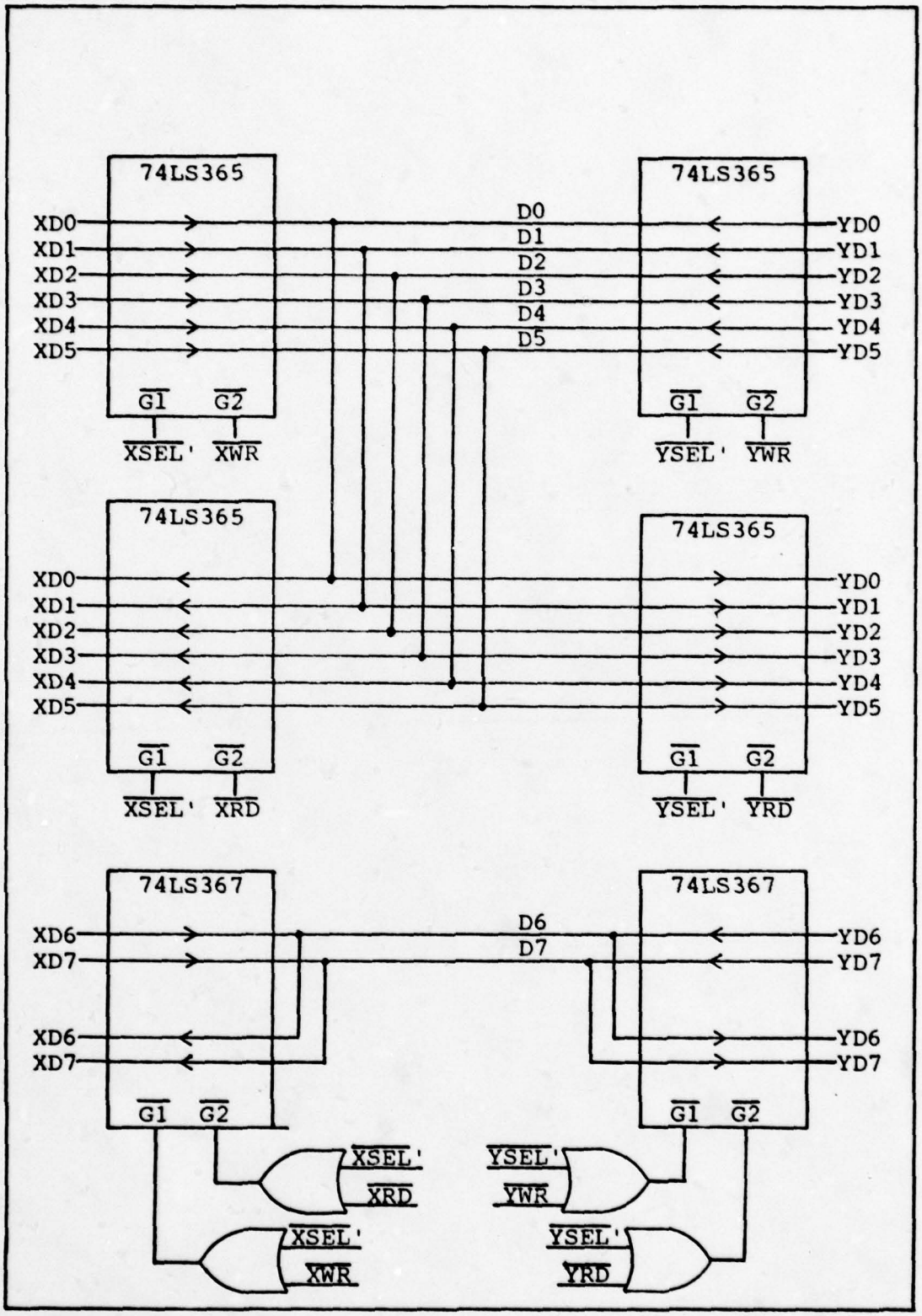


Fig. 2-5. Data Bus Control Signals

these signals are used to control the data buses. Basically the $\overline{\text{XSEL}}'$ and $\overline{\text{YSEL}}'$ signals act as enable signals on the X and Y data bus drives, respectively. Meanwhile, the $\overline{\text{XWR}}$, $\overline{\text{YWR}}$, $\overline{\text{XRD}}$ and $\overline{\text{YRD}}$ signals are used to control the direction in which the bus is driven.

The control signals to the shared memory and back to the two processors are $\overline{\text{WR}}$, $\overline{\text{RD}}$, $\overline{\text{MREQ}}$, $\overline{\text{RFSH}}$, $\overline{\text{XWAIT}}$ and $\overline{\text{YWAIT}}$. The $\overline{\text{WR}}$ and $\overline{\text{RD}}$ signals are managed by the $\overline{\text{STROBE}}'$ and the $\overline{\text{XSEL}}$ signals as indicated during the discussion relating to the control of the address bus. Figure 2-6 shows how various arbitration logic signals are combined to produce $\overline{\text{MREQ}}$, $\overline{\text{RFSH}}$, $\overline{\text{XWAIT}}$ and $\overline{\text{YWAIT}}$. $\overline{\text{MREQ}}$ is developed by delaying the logical AND of $\overline{\text{XSEL}}'$, $\overline{\text{YSEL}}'$, $\overline{\text{XRFMQ}}$ and $\overline{\text{YRFMQ}}$. The delay is necessary to insure that the addresses going to the shared memory have had time to stabilize before being gated by the $\overline{\text{MREQ}}$ signal. The $\overline{\text{RFSH}}$ signal is developed from the logical AND of the $\overline{\text{XRFSH}}'$ signal and the $\overline{\text{YRFSH}}'$ signal. $\overline{\text{XWAIT}}$ is generated when $\overline{\text{XSEL}}$ goes low at a time in which the Y processor is performing a memory operation (XEN is low). Conversely $\overline{\text{YWAIT}}$ occurs when $\overline{\text{YSEL}}$ goes low during a X processor memory operation (YEN is low).

This completes the discussion of the signal flow in the Dual Processor Card. It should be evident from the discussion that it is possible to lose refresh cycles in the implementation of the arbitrator. Preliminary analysis indicates that the loss of a given refresh cycle would not

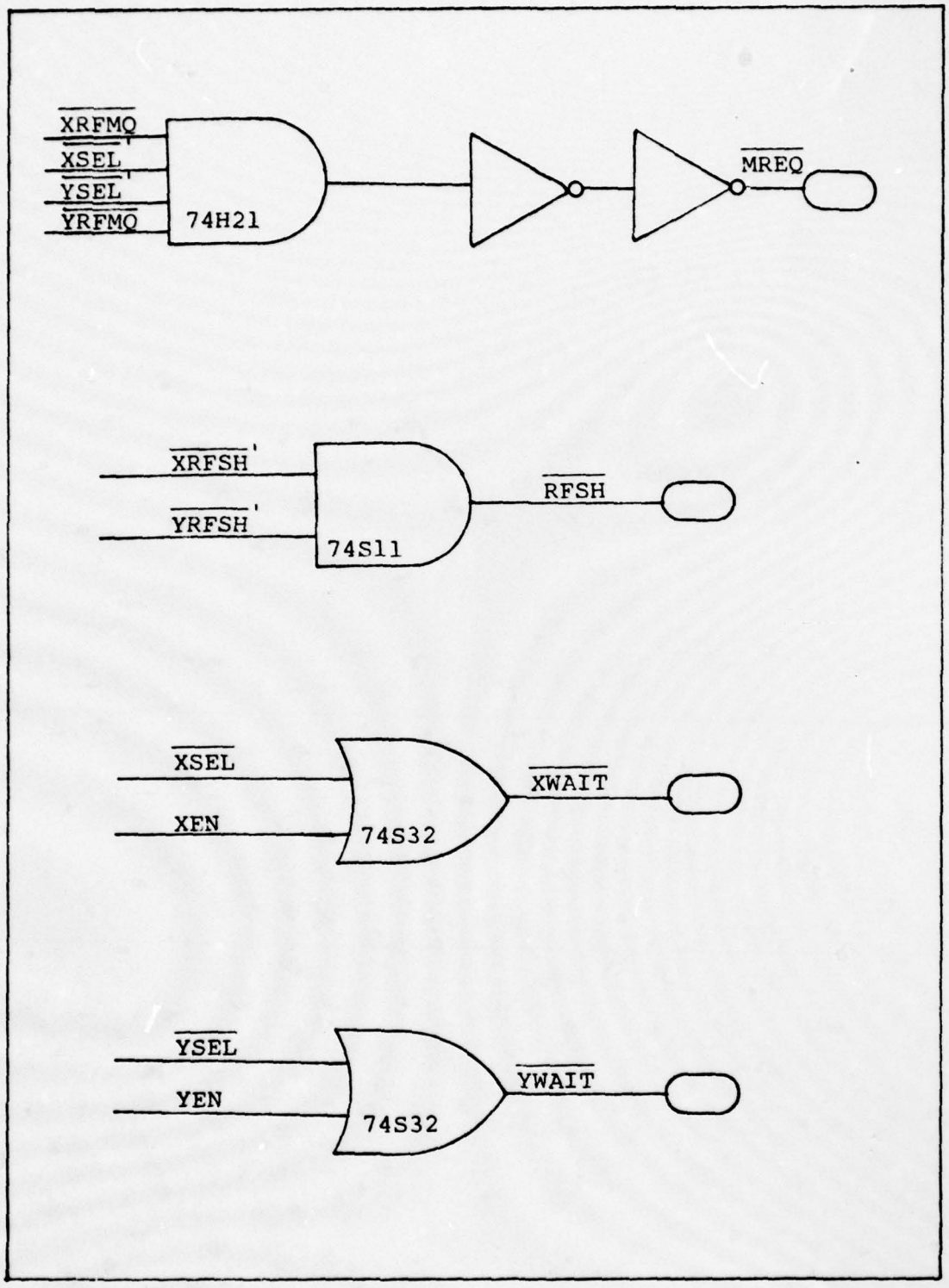


Fig. 2-6. $\overline{\text{MREQ}}$, $\overline{\text{RFSH}}$, $\overline{\text{XWAIT}}$ and $\overline{\text{YWAIT}}$ Signals

cause the loss of any data. In a worst case analysis; however, it was found that 128 refresh cycles will occur at least every 820 μsec, if the processor is decoding and processing single op code 16-bit loads (for example LD HL, (nn)). This instruction requires 16 clock cycles for decoding and execution (6.4 μsec for a 2.5 MHz execution time). Thus, in this case there would be at least two more chances to perform the refresh before the 2 msec refresh time specification was surpassed. It is unlikely that this would occur. Also it should be noted that there is a second processor providing refresh signals and that the typical time between individual refresh cycles is about 7 clock cycles or 2.8 μsec. For these types of instructions, to complete 128 refresh cycles would require around 360 μsec which indicates over 5 complete refresh cycles every 2 msec. Therefore, the loss of some refresh cycles should not cause any loss of data.

As stated before, a complete schematic is included in Appendix A. The Dual Processor Card was implemented on a standard Z-80 Wire Warp Board which is the same size as the other Z-80 boards. These wire wrap boards include space for sixty 16-pin devices along with several 24, 28, or 40-pin devices. In the schematics the pin numbers correspond to the pin numbers of the sockets rather than the pin numbers of the 14-pin devices used.

Summary

The Dual Processor Card arbitrates memory accesses and the signals which provide for the memory refresh. The arbitration logic selects which memory access or memory refresh operation will take place and locks out other requests until the operation is completed. The bus and control signal logic uses signals from the arbitration logic and from both processors to generate control signals and to control the flow of data on the address and data buses.

The design of these two parts of this Card are not unique. It would be possible to implement the functions performed by the Card using a different design. It would be possible to use one processor as the only source for the refresh signals and to place the second processor in a mode such that it could make memory accesses on a non-interference basis. A design such as this would increase average access time for at least the second processor, even so, the configuration could be implemented. Other alternatives are feasible, but allowing for the shared memory to be accessed on a first-come first-served basis appeared to be the best alternative.

III. Local Card

The Local Card interfaces with and is controlled by the X processor. It was designed to provide four RS-232C compatible input/output ports to interface with either Data Communications Equipment (DCE) or Data Terminal Equipment (DTE).

This chapter will describe the circuitry of the Local Card. Also the unusual features of the Card will be discussed and how the features can be modified to provide greater flexibility in the Card. The operational aspects which must be considered when using the Card will be presented. The summary section will review the material discussed and suggest several alternatives.

Local Card Circuitry

This section addresses the various parts of the circuitry on the Local Card. The discussion on these parts will be in the following order: address and control signal drivers, data bus drivers, address decoding, Counter Timer Circuit (CTC) signals, 2651 Programmable Communications Interface (PCI) signals and interrupt handling. There is a certain amount of interdependence of signals so at certain times references to signals will be made, which have not yet been explained. This discussion will attempt to minimize this occurrence.

Address and Control Signal Drivers. All the address

lines and control signals to and from the X processor board are buffered through Hex Bus Drivers. These signals and their directions are illustrated in Figure 3-1. The drivers which are used are 74LS367 Hex Bus Drivers and both control signals are tied to ground, since the signals are continuously driven. The A7 - A0 signals are used in the address decoding process. The $\overline{WR}$ signal is inverted and used to control the direction of data flow through the bi-directional data bus buffers. The $\overline{RD}$ signal is applied to the CTC's and to the 2651 PCI's and is used when writing to and reading from those devices. The $\overline{IORQ}$ and $\overline{MI}$ signals are used in the address decoding circuitry and in the interrupt handling circuitry. The Interrupt Enable Input (IEI) and Interrupt Enable Output (IEO) are used in the interrupt handling circuitry and in controlling the interrupts initiated by other Local Cards. The $\overline{CLK}$ signal is inverted and applied to the 8214 Priority Interrupt Control Unit (PICU) and the CTC's. The CLK/2 signal is applied to the external timer input of each of the four CTC channels, as part of the Baud Rate generation process. The circuitry associated with the data bus drivers will be covered next.

Data Bus Drivers. Two Intel 8216 4-Bit Parallel Bi-Directional Bus Drivers were chosen as the drivers of the data bus. Figure 3-2 shows the circuitry and signals immediately a part of the data bus and its controlling

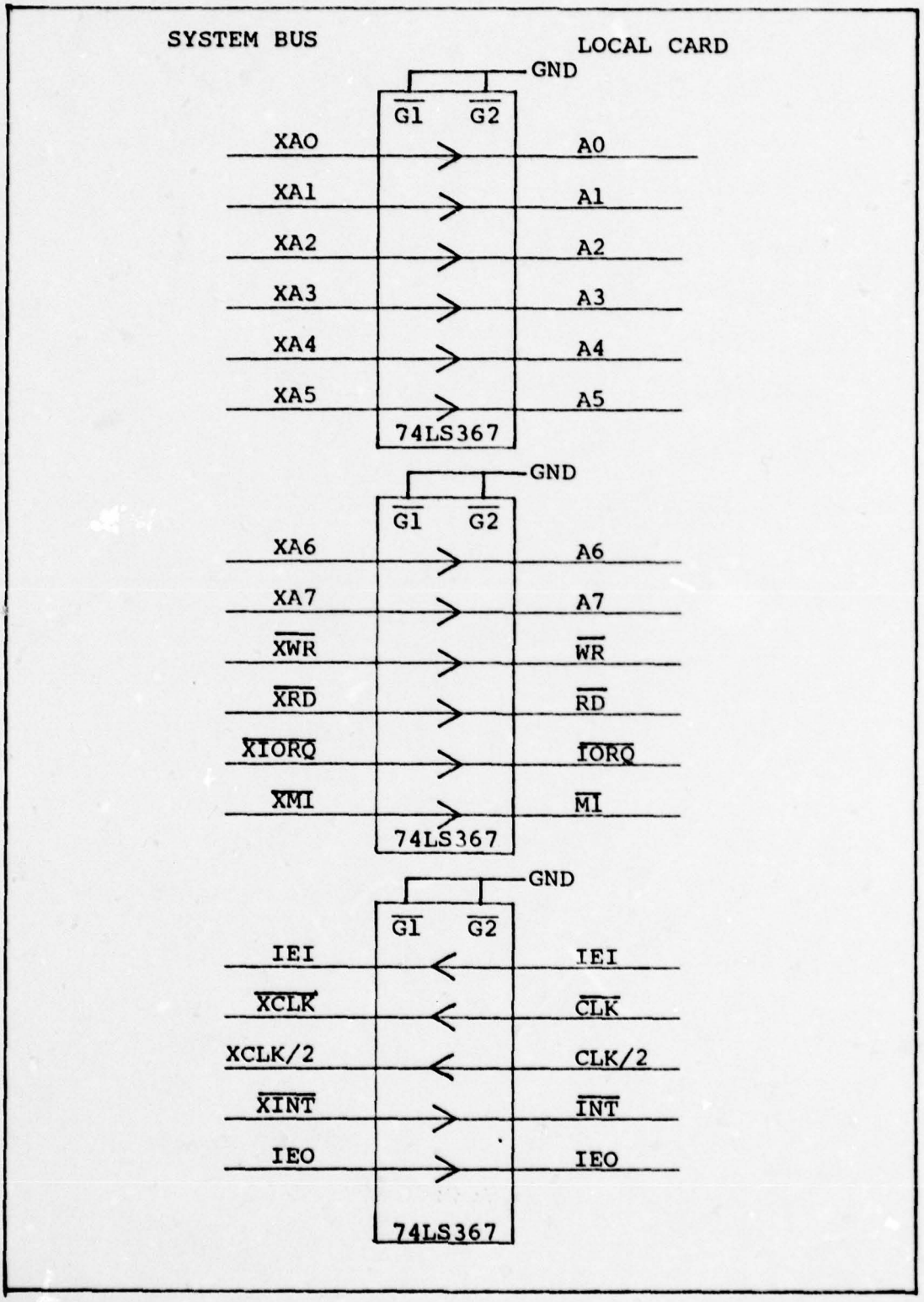


Fig. 3-1. Address and Control Signal Drivers

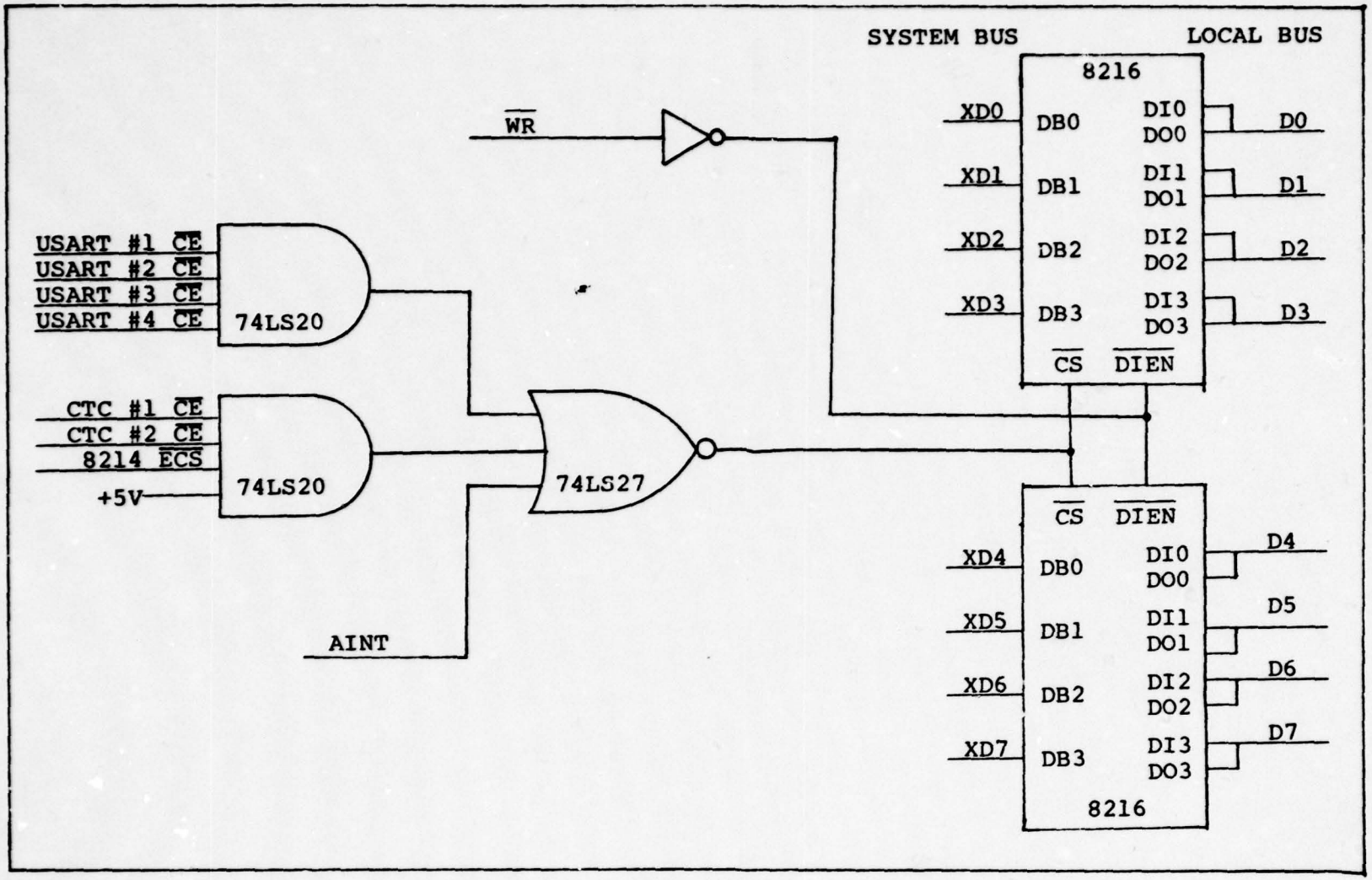


Fig. 3-2. Data Bus Drivers and Controlling Circuitry

circuit. The data bus side of each device is connected to the X processor data bus, while each data in/data out pair had their pins connected which were in turn connected to each CTC and to each 2651 PCI to allow data transfer to these devices. As stated before the $\overline{WR}$ signal is inverted and is applied to the $\overline{DIEN}$ control to define the direction the data bus is driven. Figure 3-2 also shows that if any of the CTC, the 2651 PCI or the 8214 PICU is enabled with a low signal or if the AINT (acknowledge interrupt) is high then the chip select for both data bus drivers will be driven low, thus enabling the drivers. This completes the discussion on the data bus drivers, while the next subsection deals with the address decoding circuitry.

Address Decoding. A 74LS138 3-to-8 Line Decoder is the principal device used in the decoding of address lines A7 - A0. Each Local Card requires that 25 addresses be decoded for addressing the various devices on the Card. Each of the CTC's and the 2651 PCI required four addresses, while the 8214 PICU in the interrupt handling circuitry required one address. Figure 3-3 shows the circuitry involved in the address decoding process. Address lines A1 and A0 were applied to the CTC's and the 2651 PCI as channel selects and internal register selects, respectively. Address lines A4 - A2 were applied to the 3 select inputs of the 3-to-8 Line Decoder to develop the true low chip

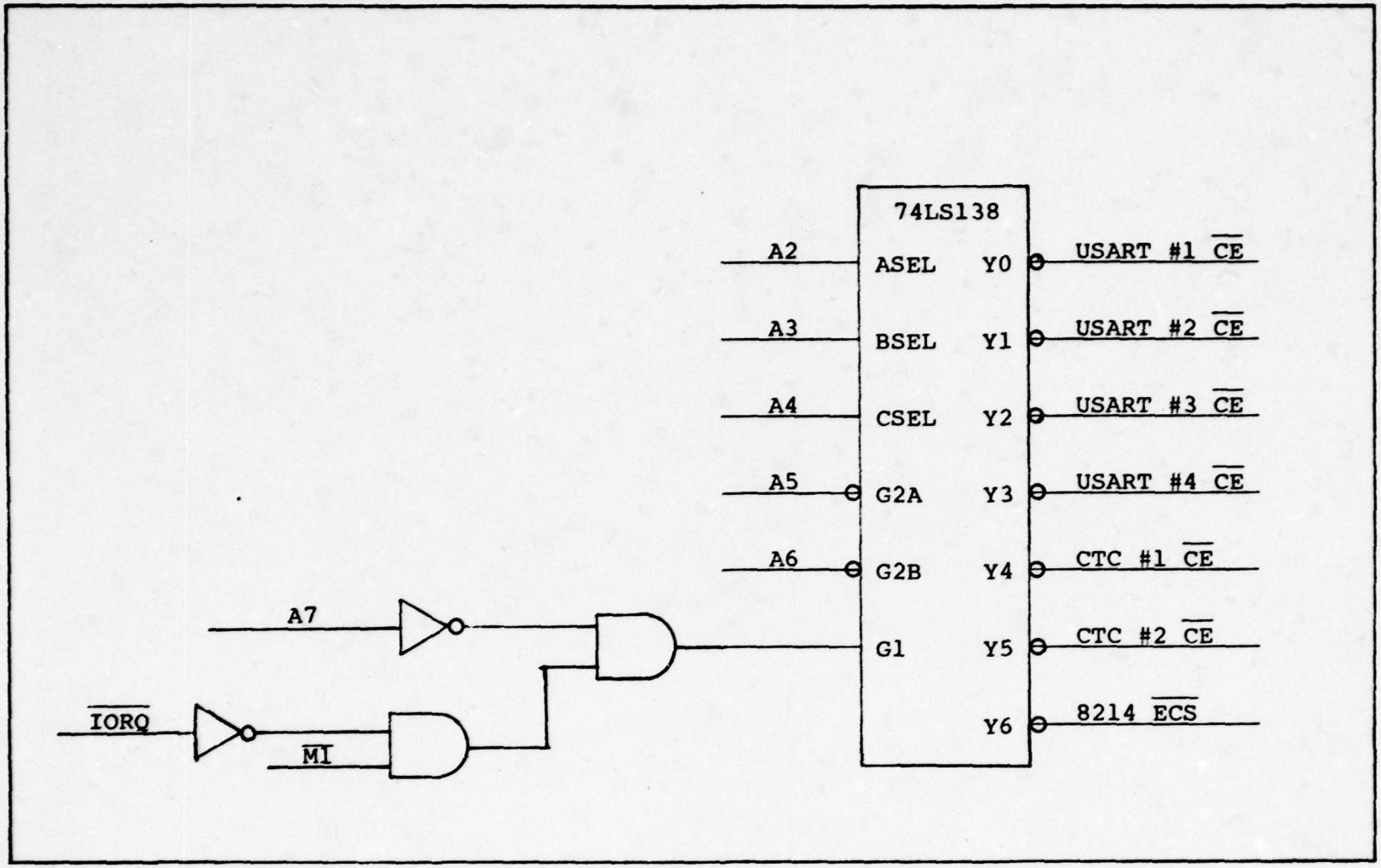


Fig. 3-3. Address Decoding Circuitry

enable signals for the CTC's, the 2651 PCI's and the 8214 PICU. Address lines A7 - A5 are applied to the 3-to-8 Line Decoder enables as required to have the Decoder respond to a certain address space. Since the Local Card requires 25 unique addresses and it takes, at least, 5 address lines to generate those addresses, then by properly wiring address lines A7 - A5, 8 separate Local Cards can be supported under the Z-80 Input/Output addressing capabilities. In the Local Card, which was implemented, A7 - A5 were set to place the address space of the Card in the lowest eighth of the address space of the eight address lines. Finally, since the $\overline{\text{IORQ}}$ is also active during the interrupt acknowledge cycle, the $\overline{\text{M1}}$ signal is logically combined as shown to insure that the Decoder only operates during an input or ouput cycle. This concludes the discussion relating to the address decoding circuitry. The next subsection deals with the CTC.

Counter Timer Circuit. The Counter Timer Circuit is used with the internal baud rate generation capabilities of the 2651 PCI to provide wide flexibility in the choice of baud rates under which the 2651 will operate. Figure 3-4 shows which signals are applied to each of the two CTC's. Appropriately the CTC's are designated #1 and #2. The signals shown in the Figure are for CTC #1. The signals for CTC #2 are the same except for the chip enable and the zero timeout pins. CTC #1 provides the timing for

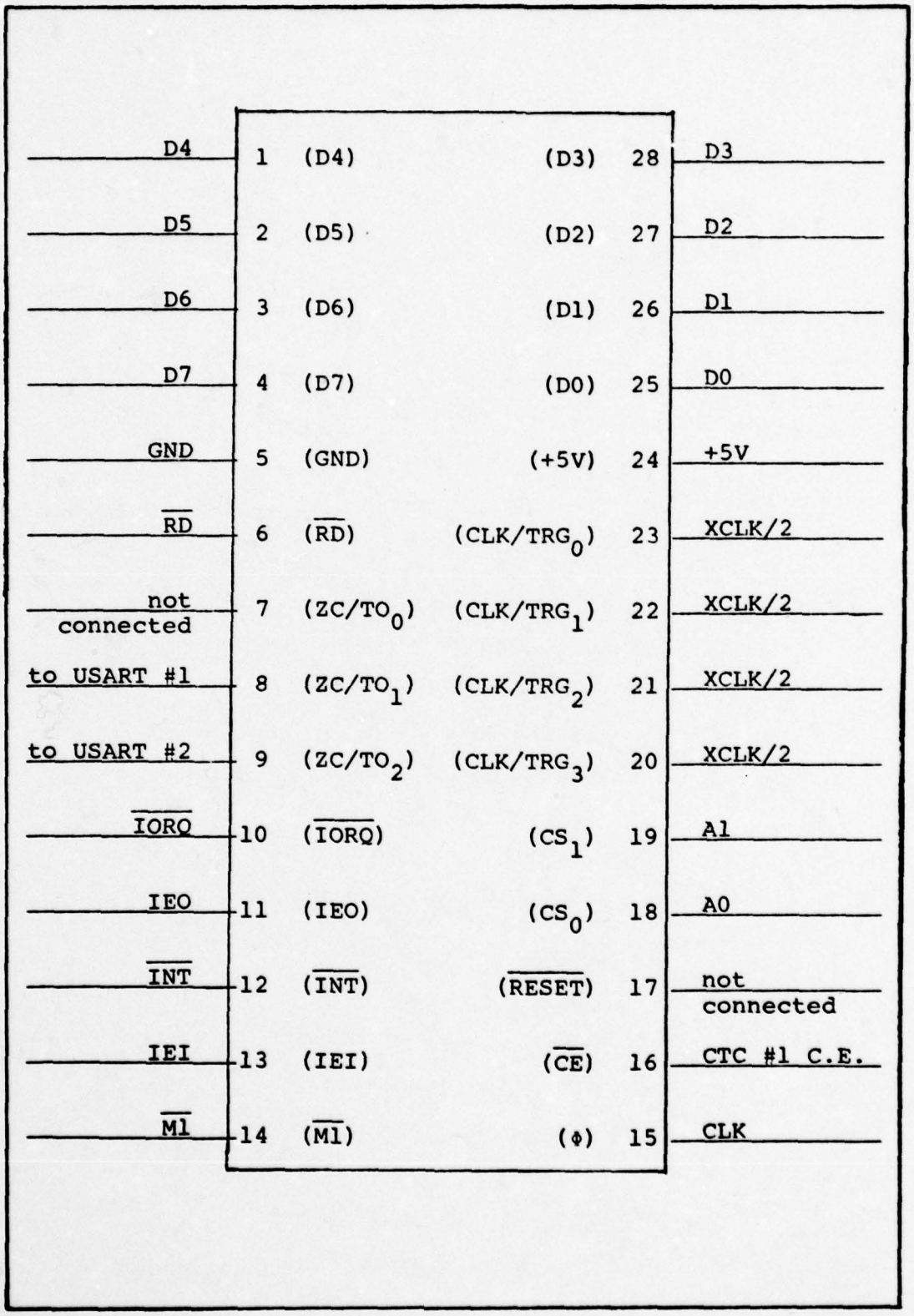


Fig. 3-4. CTC #1 Signals

the first two of the 2651 PCI (henceforth, called USART's), while CTC #2 provides the timing signals for USART's #3 and #4. The Figure provides the number and name of each of the 28 pins of the CTC along with the name or destination of the signal at that pin. Therefore, the Figure should be self-explanatory so that discussion will continue with the 2651 Programmable Communications Interface.

2651 Programmable Communications Interface (PCI). The 2651 PCI was chosen as the Universal Synchronous/Asynchronous Receiver/Transmitter (USART) for the Local Card. Four USART's are used on each card to provide four full duplex RS-232C channels. Each of these USART's use almost the same set of signals with the exception of the chip enable, baud rate clock input, $\overline{\text{TxRDY}}$, and $\overline{\text{RxRDY}}$. The sources and destinations for these four pins vary slightly among the four USART's. Figure 3-5 illustrates the signals being applied to USART #1. The Figure provides the name and number of each pin on the chip along with either name, source or destination of the signal at that pin. This Figure is also self-explanatory, so that the discussion will continue on to the interrupt handling circuitry.

Interrupt Handling. After the 2651 PCI has completed a data transfer in or out, an interrupt will be issued by pulling the $\overline{\text{TxRDY}}$ or the $\overline{\text{RxRDY}}$ low. Since there are four of these USART's on the Card, it is necessary to prioritize the interrupts and develop the vectored interrupt address. These functions are performed through the use of the Intel

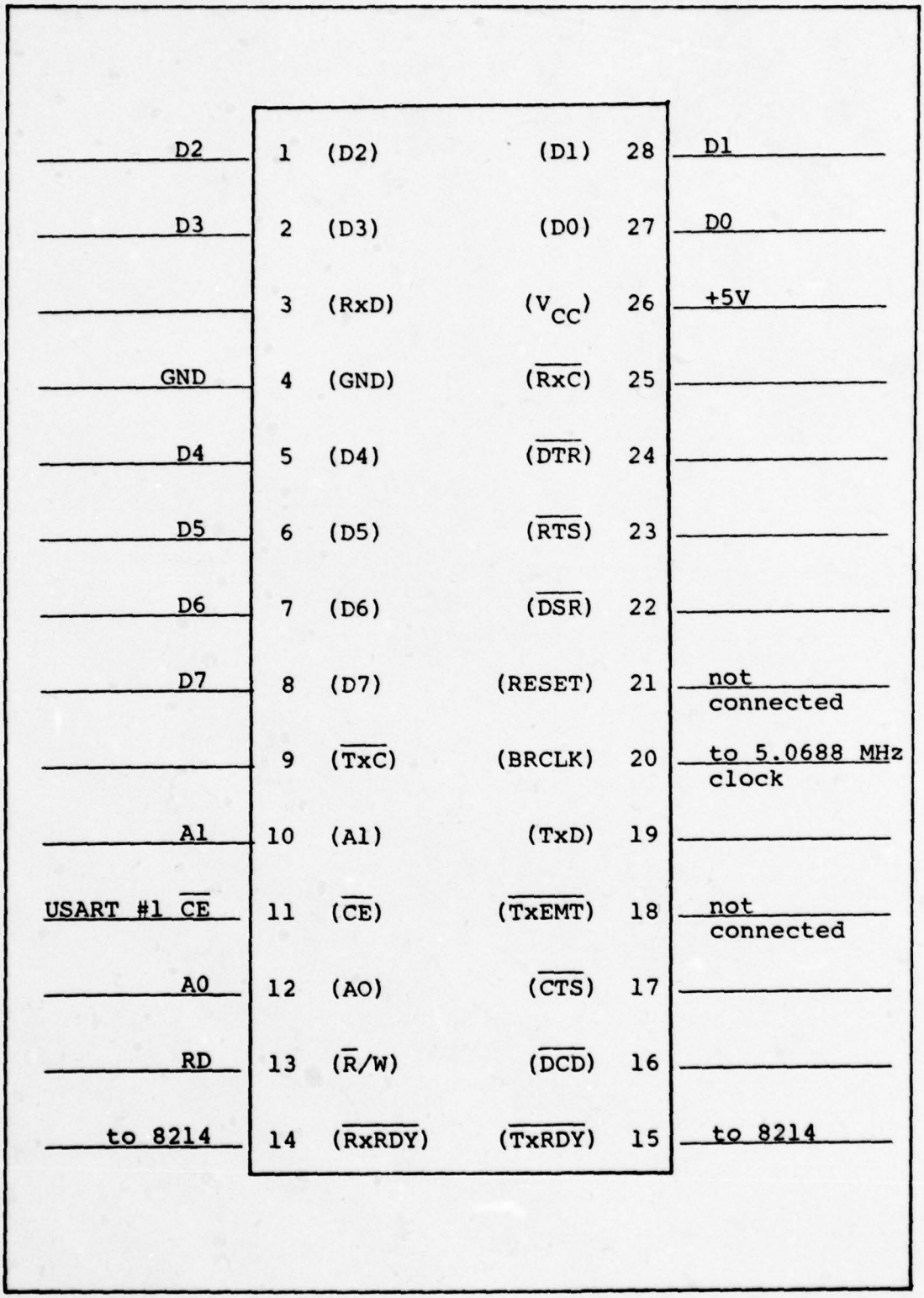


Fig. 3-5. USART #1 Signals

8214 Priority Interrupt Control Unit, the Intel 8212 Eight-Bit Input/Output Port and several simple gates. Figure 3-6 is a schematic of this interrupt handling circuitry. The 8214 PICU can prioritize among eight different interrupts. The request lines are designated $\overline{R7}$ - $\overline{R0}$ where an interrupt on $\overline{R7}$ has the highest priority. The $\overline{RxRDY}$ signals of USART's 1 - 4 are input at $\overline{R7}$ - $\overline{R4}$, while the $\overline{TxRDY}$ signals are input at $\overline{R3}$ - $\overline{R0}$. Thus, USART #1 $\overline{RxRDY}$ has the highest priority, while USART #4 $\overline{TxRDY}$ has the lowest priority. The Status Group Select ($\overline{SGS}$) signal and the $\overline{B2}$, $\overline{B1}$, and $\overline{B0}$ inputs are used to enable various priority levels and above; however, since the Card requires the eight interrupt lines, $\overline{SGS}$ is tied high enabling all interrupts. The Master Interrupt Enable (INTE) and the Enable This Level Group (ETLG) pins are tied to the Interrupt Enable Input (IEI). The Enable Next Level Group (ENLG) becomes the Interrupt Enable Out (IEO) for this Card. Also, the Enable Level Read ($\overline{ELR}$) is tied low to allow reads at this chip at any time. The PICU is enabled by writing to its port address. In this particular design, an OUT instruction to address $000110XX_2$ would provide the required Enable Current Status ($\overline{ECS}$) signal. After the 8214 has been enabled, the highest prior interrupt pending or the first interrupt to the device would cause the 8214 to issue an interrupt ($\overline{INT}$) and encode the priority level of the interrupt on signals $\overline{A2}$, $\overline{A1}$, and $\overline{A0}$. With the Mode set low and Device

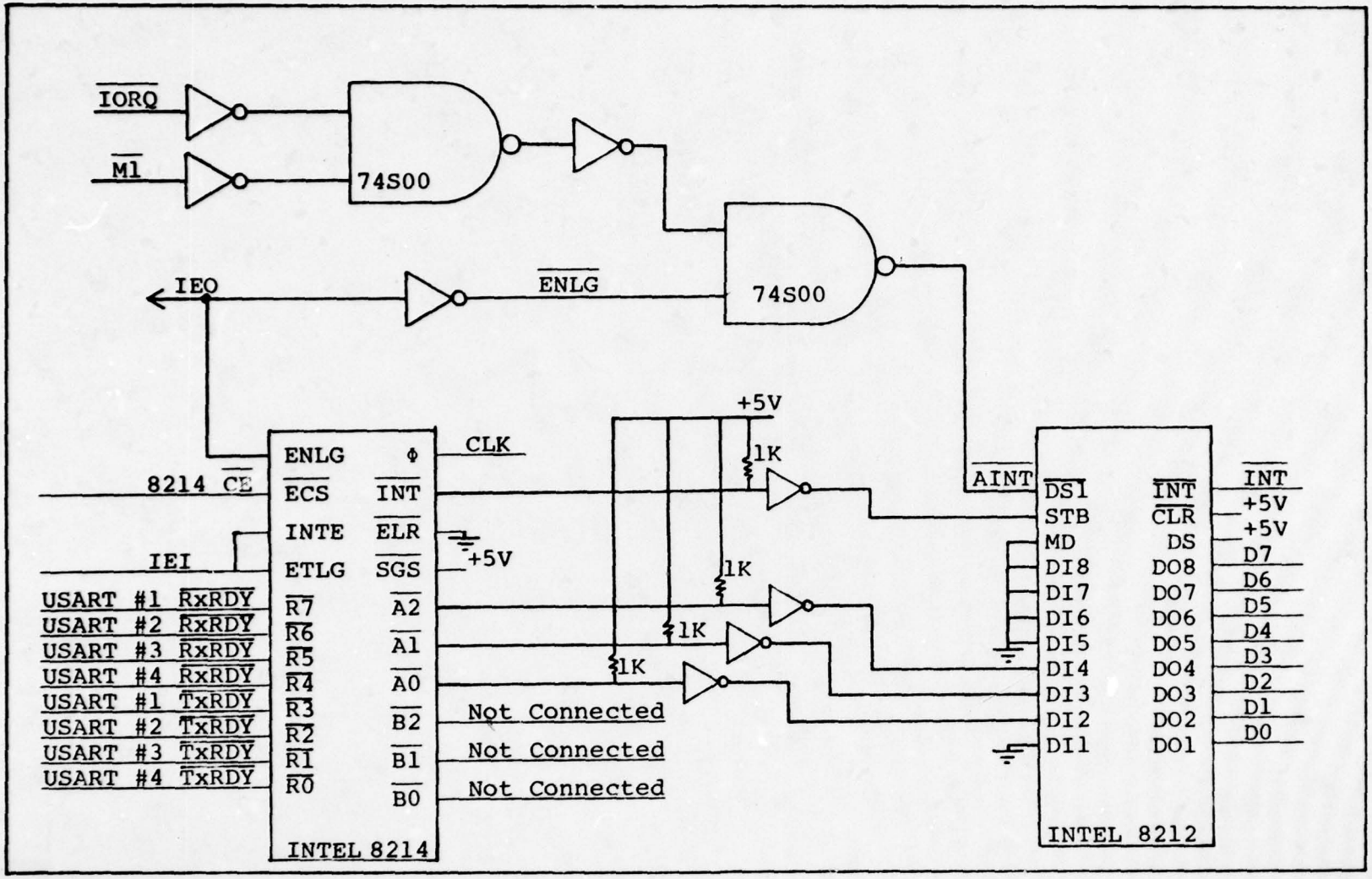


Fig. 3-6. Interrupt Handling Circuitry

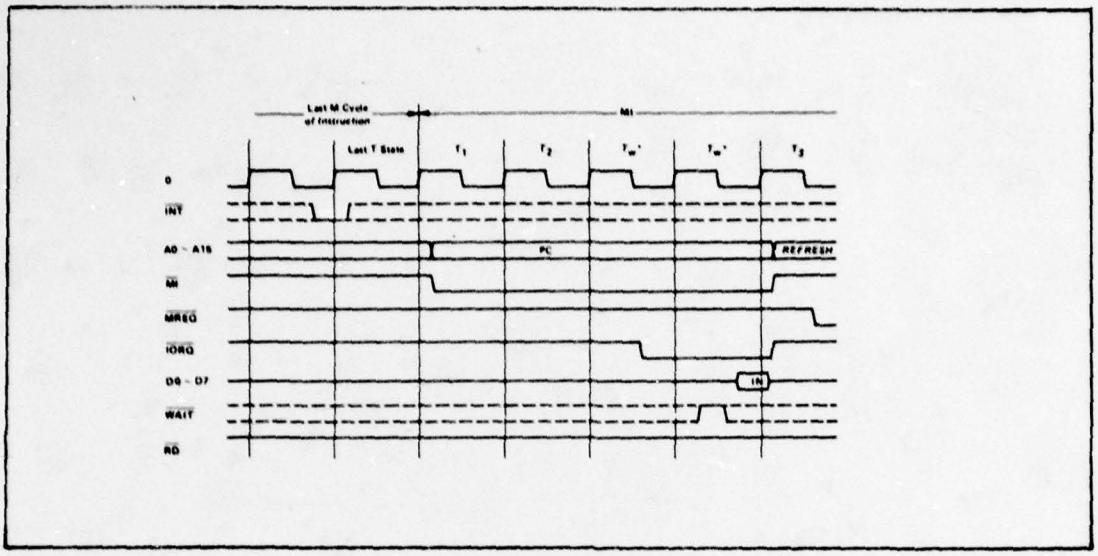


Fig. 3-7. Interrupt Request/Acknowledge Cycle (Ref 17)

Select 2 set high, the data from the 8214 is gated into the appropriate latches after being inverted. The configuration as shown set DI8, DI7, DI6, DI5, and DI1 to 0, while DI4 = A2, DI3 = A1, and DI2 = A0. When an interrupt acknowledge is recognized and processed, then the interrupt vector will be 0000XXX0 corresponding to an even address in the jump table for Z-80 vectored interrupts. The XXX will be 111 for an $\overline{R7}$ interrupt and 000 if the interrupt had been an $\overline{R0}$.

The interrupt acknowledge signal is developed from the $\overline{IORQ}$ and $\overline{M1}$ signals of the microcomputer and the Enable Next Level Group (ENLG) signal from the 8214. Figure 3-7 shows how the CPU responds to an interrupt. A special $\overline{M1}$ cycle is generated with two wait states and $\overline{IORQ}$ goes low. These signals are inverted and ANDed together and then this

result is NAND'ed with the inverse of ENLG. ENLG will go low when the 8214 generates an interrupt. The result is the $\overline{\text{AINT}}$ which is input to the Device Select 1 (DS1) causing the interrupt vector to be put out on the data bus. Then, the interrupt handling circuitry is re-enabled by writing to the 8214 Enable Current Status ($\overline{\text{ECS}}$) pin. This concludes the discussion on the interrupt handling circuitry. A complete schematic is provided in Appendix B. Inspection of the schematic will reveal the existence of line drivers and receivers associated with each USART. Discussion of these drivers and receivers is delayed until the features section of this chapter because the configuration of the drivers and receivers is dependent on the requirements of the port. Finally, it is recognized that the design of some of the circuitry is inefficient, this is primarily due to the availability of chips and the desire to get the device working first and efficient later. Alternatives to the design are provided in the summary section.

Features of the Local Card

It was stated that more than one Local Card can be used in the Universal Network Interface Device. This is desirable to allow for adding capability to the system without requiring a complete re-design. The modular nature of the Local and Network Cards makes this idea very practical. To add more cards, it would be necessary to perform

four options. These options would be: 1) correctly jumper the address decoding circuit of additional cards to preclude any conflicts when performing I/O addressing to each card; 2) correctly jumper pins DCE - DTE on the RIU to provide unique vectored addresses for the interrupts; 3) establish which card will have the greatest priority then use the Daisy-Chain Enable Out (DEO) from that card as the Interrupt Enable In on the next card and so on; and 4) modify the I/O assembly software as necessary to handle the additional cards.

One feature of the card is that the I/O ports are RS-232C compatible. This specification defines the transmission and reception of data between Data Communication Equipment (DCE) and Data Terminal Equipment (DTE). DCE is a term used to describe communication devices used to transmit the binary data generated by a computer. DTE refers to the computers or terminals which are the source and final destination for the transmitted binary data. In order to allow a given port to interface with either DCE or DTE it is necessary to be able to configure the port as DTE or DCE respectively. The RS-232C includes much flexibility as to how the signals are generated, particularly clocking signals between the DTE and the DCE. In the configurations, shown in Figures 3-8 and 3-9 it will be assumed that the DCE is the source for the receiver clock and the DTE is the source of the transmitter clock. The drivers used are Motorola MC1488's and the

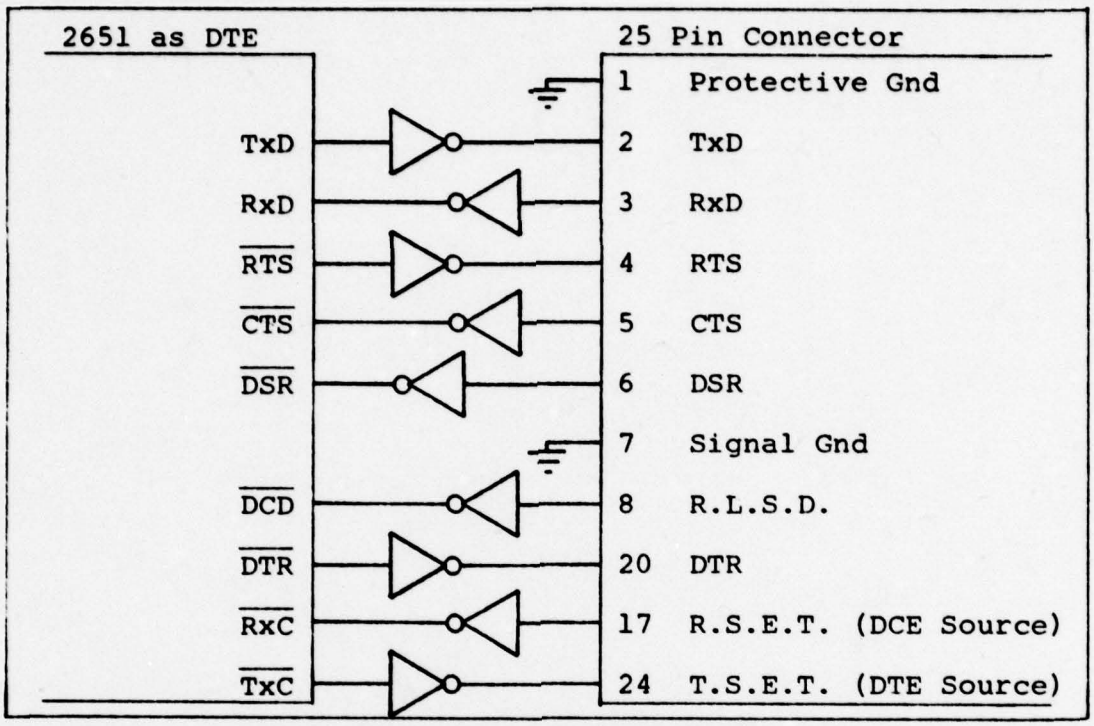


Fig. 3-8. Port Configured as DTE

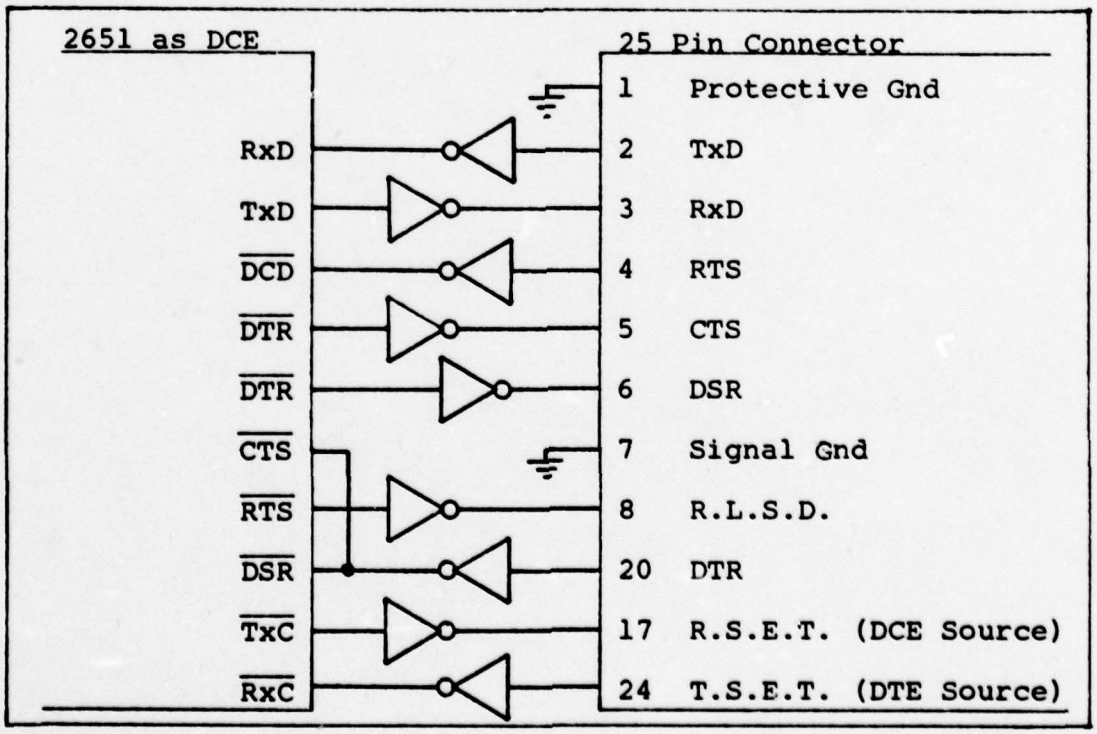


Fig. 3-9. Port Configured as DCE

receivers are Motorola MC1489's. These devices are in common use in the application of RS-232C interfaces. Figure 3-8 shows the signal arrangement for the port configured as DTE and Figure 3-9 shows the signal arrangement for the port when it is configured as DCE. One point to note is that the connector used would be a standard 25-pin female RS-232C connector. Also the Card could be configured to support the RS-232C secondary channel through the use of two ports instead of just one. This concludes the discussion on the features of the Local Card. The last section of this chapter will present an overview of the operations on the Local Card.

Local Card Operations

This section addresses how the Local Card is used and what steps must be performed to program the devices. Depending on how timing signals are supplied, the USART will determine the programming of the CTC's. If the CTC's will supply the timing signals, then they must be programmed, as follows. The CTC and channel are selected by writing out the address on address lines A7 - A0. The operating mode is selected, in this case, timer mode with prescaler and appropriate triggering to start the timing cycle. Then a time constant can be loaded as the next word written to the channel. These operations would be done for each channel of the two CTC's providing timing signals to the USART's. After the CTC's have been programmed, then the

USART's would be programmed. Details for programming these devices can be found in the appropriate Signetics product specifications; however, to give some appreciation of the devices the following discussion is presented. Assuming that the ports are supporting asynchronous communication then the first step would be to load the settings into the two mode registers. This would be accomplished by outputting address 00000001 for USART #1 and selection of the appropriate settings to specify number of stop bits, parity, character length, mode and baud rate factor. Outputting the same address would allow the second mode register to be loaded. The second mode register controls which clocks are used and selects the baud rate. Then, the command register is loaded by writing out to address 00000011. The command register is used to set operating mode, Request to Send ($\overline{\text{RTS}}$), Data Terminal Ready ($\overline{\text{DTR}}$), reset receive control and transmit control. The last two items are used to enable the receiver and transmitter, respectively. If both are set, then the receiver will receive data when the $\overline{\text{DCD}}$ input is low and the transmitter will transmit data when the $\overline{\text{CTS}}$ input is low. After the devices have been initially programmed and the support software is available then a write to the 8214 Enable Current Status would allow the Card to process interrupts and to transfer data. This concludes the description of the operational features of the Local Card.

Summary

This chapter was intended to provide the reader with a detailed review of the circuitry and operation of the Local Card. The material in the chapter dealt with the buffering, port decoding and interrupt handling circuitry. The designs used for these three types of circuitry can be modified rather easily. The extensive resources available to the digital designer allow many different ways for implementing different functions. The designs selected for these three circuits were chosen on the basis of simplicity and flexibility. The CTC and 2651 PCI perform more complicated functions; however, there are other devices available which perform similar functions. The application of control signals would be different, if other devices were used, but not greatly. Therefore, the design of this Card could be changed, yet still accomplish the required functions.

IV. Network Card

This chapter deals with the Network Card in a manner similar to the previous chapter. The various parts of the circuitry will be described as they fit within the following function: address and control signal drivers, data bus and the associated control circuitry, address decoding, the Counter Timer Circuit and the Z80-Serial Input/Output device. The jumpering options and the configuration of the port are also discussed. A brief discussion on the Network Card operations is then presented. The last section summarizes the chapter and discusses alternatives.

Network Card Circuitry

This section provides a detailed description of the circuitry of the Network Card. The design is very similar in general appearances to that of the Local Card. There are, however, several aspects to the two cards where different designs were implemented due to differences in the modes of operations. The first part of the circuitry to be covered is the address bus and control signal drivers.

Address Bus and Control Signal Drivers. Using the same design, as was used on the Local Card, the address lines and control signals are buffered by using 74LS367 Hex Bus Drivers. These drivers are used to minimize loading on the system buses and to insure high quality signals

are returned to the Y processor. Figure 4-1 illustrates which signals are buffered by use of the 74LS367's. The control lines are tied low to enable the drivers all the time. Address lines A7 - A0 are used in the address decoding circuitry. The $\overline{WR}$, $\overline{IORQ}$, $\overline{M1}$, and the $\overline{MREQ}$ signals are combined with other signals generated on the Card to control the data bus. The $\overline{RD}$ signal is used by the Counter Timer Circuit (CTC) and the Serial Input/Output when data is being transferred to and from these devices. The $\overline{RESET}$ signal is used by the CTC and the SIO to restart the devices. The Interrupt Enable Input (IEI) and the Interrupt Enable Output (IEO) are used in establishing a daisy chain interrupt scheme. The $\overline{CLK}$ signal is inverted and is used to provide clock signals to the CTC and the SIO. The $\overline{INT}$ signal is used by the SIO to interrupt the Y processor. And the $\overline{WAIT}$ signal is used by the two SIO $\overline{WAIT/READY}$ pins to synchronize the CPU to the SIO data rate. The next subsection will describe the data bus and the associated control signals.

Data Bus and Controlling Circuitry. Intel 8216 4-Bit Parallel Bi-Directional Buffers were selected to buffer the data bus signals. Figure 4-2 is a schematic of the circuitry used to provide the Chip Select ($\overline{CS}$) and the Data In Enable ($\overline{DIEN}$) signals to the buffers. The 8216's were wired to have the data bus pins tied to the system bus, while the data input and data output pins were connected and provided the data bus on the Card.

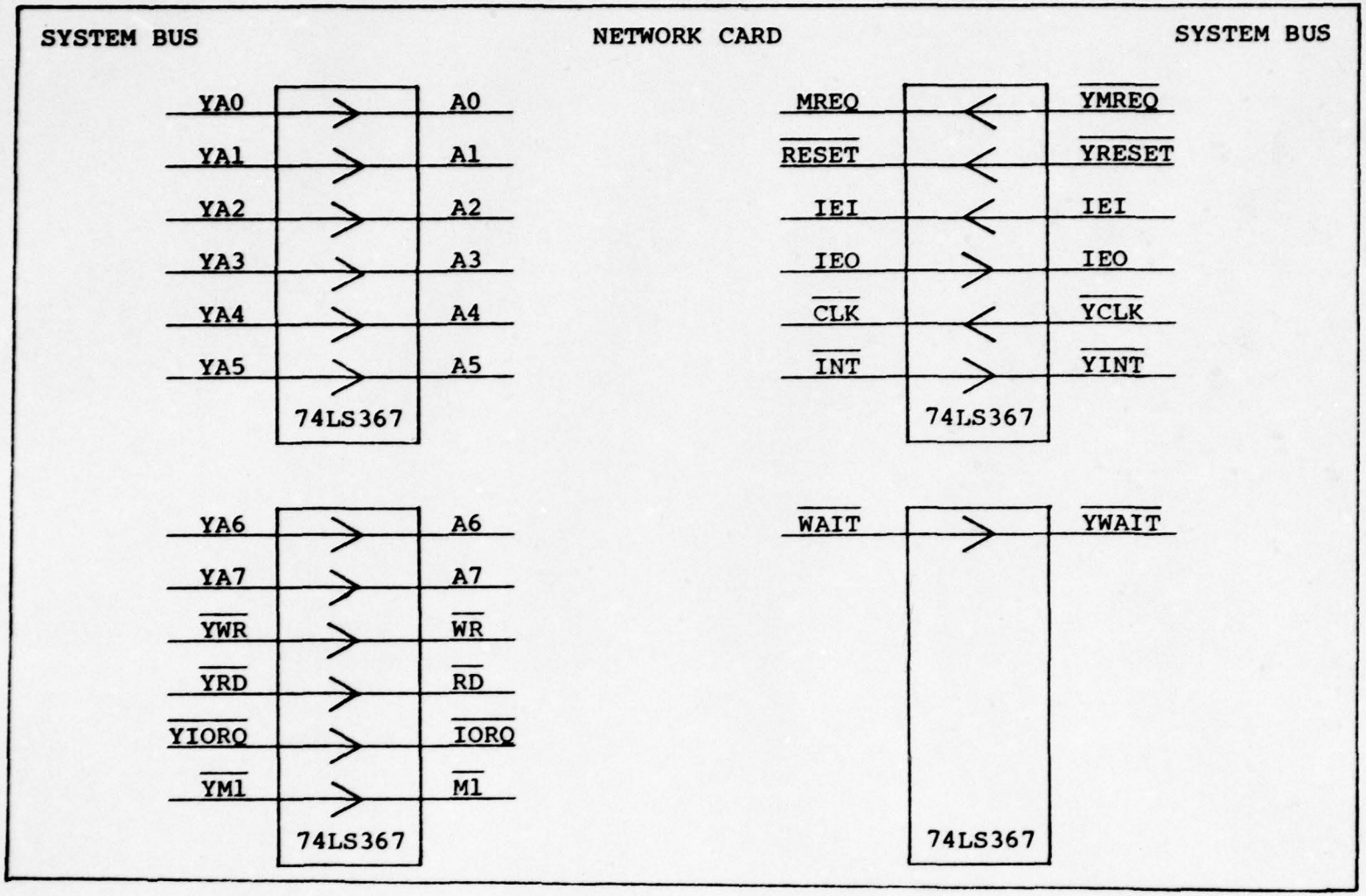


Fig. 4-1. Address and Control Signal Drivers

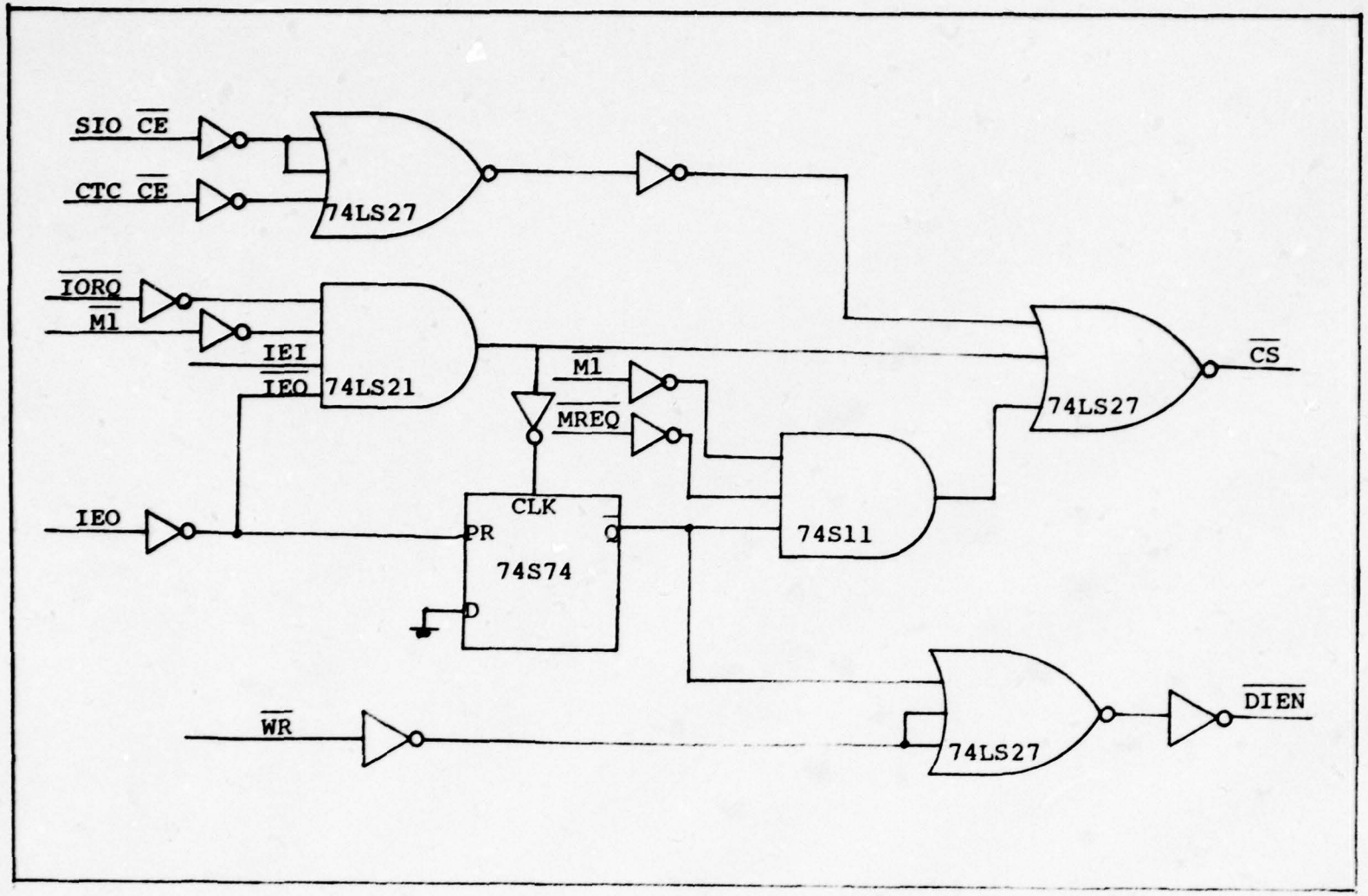


Fig. 4-2. Data Bus Controlling Circuitry

The $\overline{CS}$ signal enables the buffers, while the $\overline{DIEN}$ signal controls the direction of the data. If $\overline{DIEN}$ is high, data can be written into the Card; however when $\overline{DIEN}$ is low, data can be read from the Card. Data is transferred between the Y processor and the Network Card during reads and writes to the CTC or SIO and during the handling of interrupts. When either the CTC or SIO is enabled, the $\overline{CS}$ pin will be driven low. The $\overline{WR}$ signal is inverted and wired to an OR gate causing $\overline{DIEN}$ to be high during write operations and low during read operations. The remaining part of the circuitry shown in Figure 4-2 controls data flow during interrupts. The IEO is usually high, which causes a low at $\overline{Q}$ of the flip-flop. This low signal does not affect the data flow. When an interrupt occurs, the SIO pulls the IEO low until the service routine for the interrupt is completed and a Return from Interrupt Instruction (RETI) has been placed on the data bus and been recognized by the SIO. The interrupt from the SIO causes the CPU to initiate an interrupt acknowledge cycle. The timing of signals involved in this cycle was presented in Figure 3-7. The preset signal is removed from the flip-flop when IEO goes low, and when $\overline{M1}$ and $\overline{IORQ}$ are low a high is present at the output of the 74LS21 which causes the $\overline{CS}$ signal to go low. Meanwhile, the $\overline{Q}$ output from the flip-flop is still low, because the flip-flop will only respond to a leading edge trigger. Therefore, $\overline{DIEN}$ will be low allowing the interrupt vector to be placed on the data bus by

the SIO and to be read by the CPU. When $\overline{M1}$ and $\overline{IORQ}$ return to a high state, $\overline{CS}$ goes high and $\overline{Q}$ triggered to go high. The high $\overline{Q}$ signal will allow data to be written into the Card when $\overline{CS}$ is true. The $\overline{CS}$ will go true when both $\overline{MREQ}$ and $\overline{M1}$ are low. Thus, data is available to the SIO for decoding while instructions are being fetched from memory. The SIO must be allowed to monitor the data bus, because it recognizes the completion of its service routine by observing the two op code instruction RETI being placed on the data bus, while its IEI is high and its IEO is low. After the SIO recognizes the RETI instruction, the IEO is forced high, disabling the AND gate and presetting the $\overline{Q}$ output low. This returns the controlling circuitry to its inactive condition and allows the resumption of the other read and write control. This concludes the discussion on the data bus control circuitry. The next subsection will present the description of the address decoding circuitry.

Address Decoding Circuitry. The mode of address decoding on the Network Card is identical to that on the Local Card. The A1 and A0 signals are applied to the CTC and the SIO, while the circuitry and wiring connections shown in Figure 3-3 are used to develop the CTC and SIO Chip Enable ($\overline{CE}$). Referencing, Figure 3-3 the SIO $\overline{CE}$ is available from the Y0 pin and the CTC $\overline{CE}$ is available from the Y1 output. Therefore the addresses to the SIO and the

CTC are 000000XX and 000001XX, respectively. Other than these changes the design of the address decoding logic is identical to that described in the discussion on the Local Card circuitry. The next subsection will describe the Counter Timer Circuit.

Counter Timer Circuit Signal. The Z80-CTC is used in the Network Card to provide timing signals to the Z80-SIO. Figure 4-3 shows the signals applied to the CTC. The signals applied to the CTC are the data bus, the $\overline{CE}$, A1, A0, $\overline{M1}$, $\overline{IORQ}$, and $\overline{RD}$. These signals are also used on the CTC's of the Local Card. The timing signals input at the CLK/TRG_1 and CLK/TRG_0 pins are CLK/2 and CLK/5. These signals are generated by inverting the $\overline{CLK}$ signal and inputting the result into a 74LS90 Decade Counter. The CLK signal is applied to Input B. The Input A and Q_D pins are tied together. This configuration provides a CLK/2 signal at Q_B and a CLK/5 at Q_D. The CLK/2 signal is applied to the CLK/TRG_2 to act as a trigger signal if it is desired to use that channel. The three Zero Count or Timeout signals are made available at the $\overline{RxCA}$, $\overline{TxCA}$, and $\overline{RxTxCB}$ pins to provide a varied selection of programmable timing sources for these inputs. The system $\overline{RESET}$ signal is connected to reset the CTC during a system reset. This completes the discussion related to the CTC, the next section addresses the Z80-SIO.

Z-80 Serial Input/Output. This device provides the Network Card with its flexibility and capacity to handle

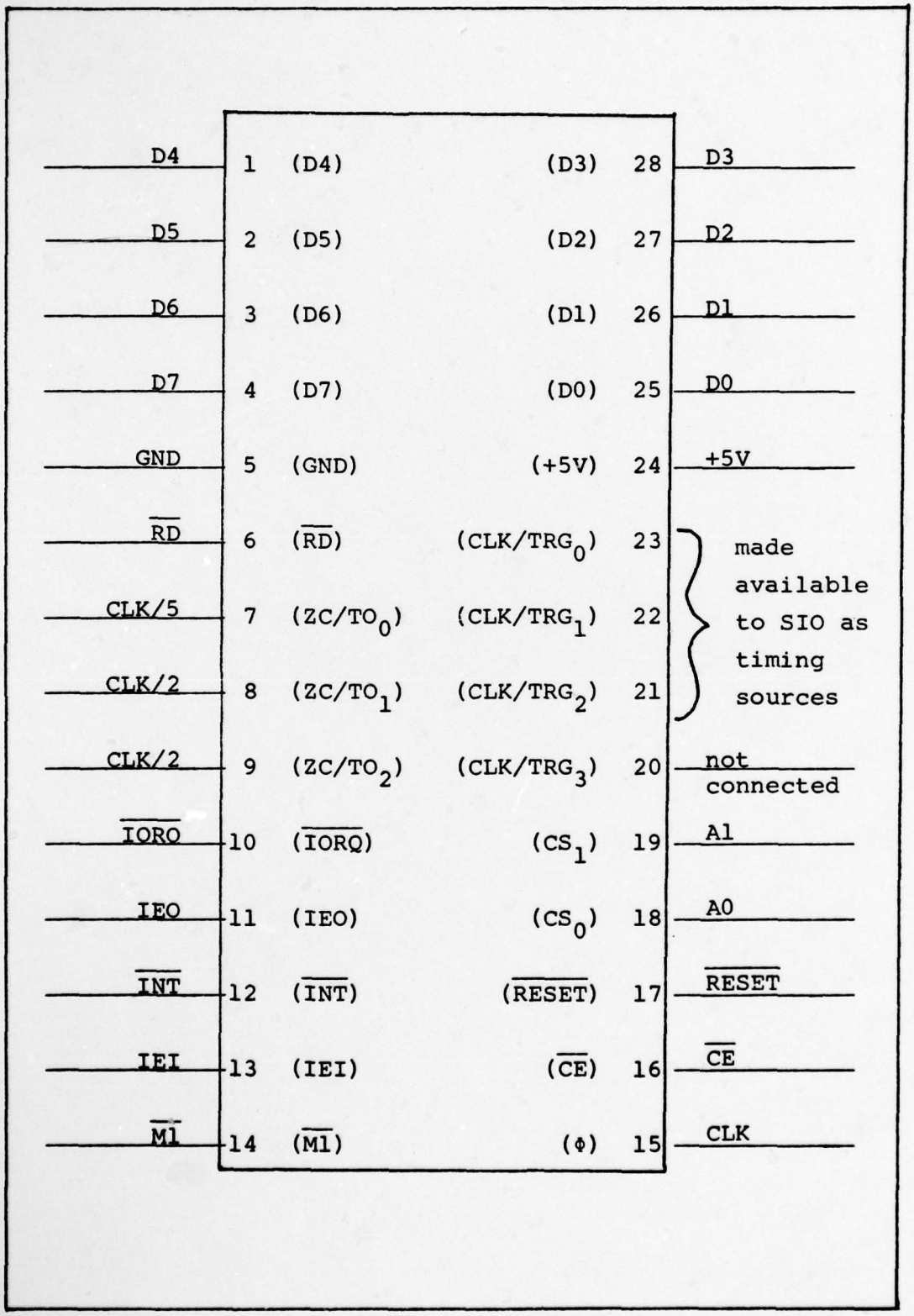


Fig. 4-3. Counter Timer Circuit Signals

high data rates and different protocols. This subsection describes the signals and the configuration of the data transmission and reception pins in order to meet the RS-422/423 specification. The dual channels of the SIO were configured such that Channel A was wired as DTE and Channel B was wired as DCE. This configuration allows testing of the Card by simulating a loop network such that messages transmitted by Channel A will be received by Channel B. Figure 4-4 provides a schematic of the signals to the SIO and the configuration of the two ports. The $\overline{W/RDY}$ signal is used to control data flow to and from the CPU. The Card is configured such that one of the CTC channels is used as a timing source for the receive and transmit clocks for both channels. The dotted lines in the drawings show which signals of each port correspond to which signals in the other port in this DTE/DCE configuration. The $\overline{SYNC}$ signals are not needed to support the Card operation as designed. This concludes the discussion on the SIO signals and the port configuration. A complete schematic is included as Appendix C. This completes the description of the Network Card Circuitry. The next section describes the features of the Card.

Features of the Network Card

The Network Card was designed to be a very flexible interface to communications networks. The flexibility of the interface is due primarily to the Z80-SIO. This

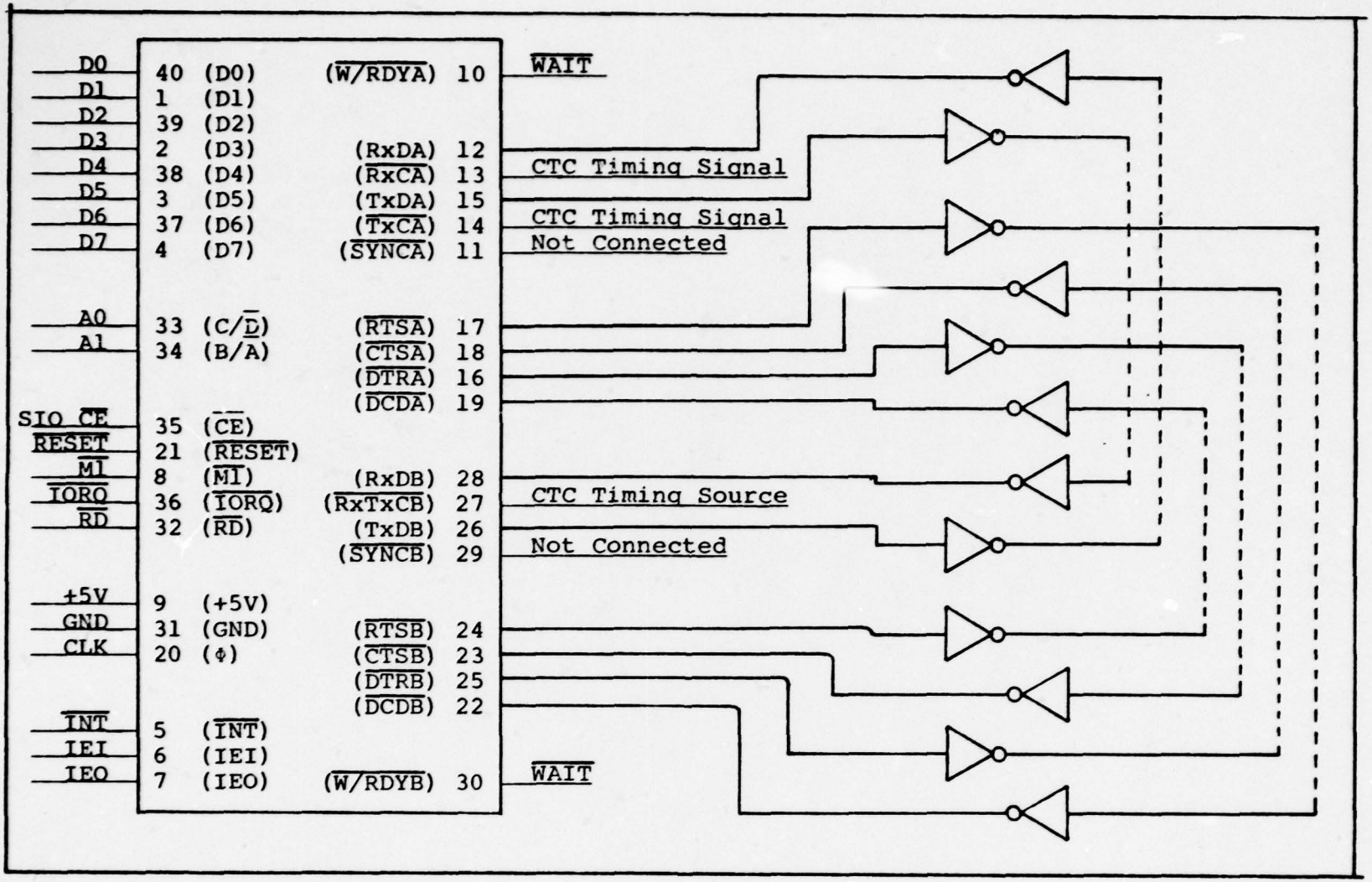


Fig. 4-4. SIO Signals and Port Configuration

device features two independent full duplex channels, data rates of up to 550 Kilobits/sec, receiver registers quadruply buffered, transmitter doubly buffered, asynchronous operation, Binary Synchronous operation, High-Level Data Link Control (HDLC) or IBM Synchronous Data Link Control (SDLC) operation, modem controls, error checking, and daisy chain priority interrupt logic including interrupt vectoring. More data on this device can be obtained from the Z80-SIO Product Specification (Ref 18). Data on the above mentioned Data Link Controls is available in references 15 and 16. Several Network Cards can be supported in the UNID; however, the high data rates would preclude operation at the highest of data rates. In fact, the current design will not meet the design goal of 1.5 Megabits/sec, but the 550 Kilobits/sec should be more than fast enough to demonstrate that the basic concept of the UNID design is feasible. The 1.5 Megabits/sec should be achievable with a device such as the Signetics 2652 Multi-Protocol Communications Controller (Ref 15) using direct memory access techniques to transfer the data into and out of memory. Under some applications involving the translation of data from one character code to another, the current 8-bit processors may not be able to process the data in the allotted time, then one of the new generation of 16-bit processors could be considered to support the operations on the network side of the device. This concludes the discussion on the Network Card's features, the next section

will review the operational aspects of the Network Card.

Network Card Operations

This section will provide a brief overview of the operational configuration of the Network Card. This section will specifically address programming of the CTC and the SIO. Programming of the CTC was covered in a previous chapter, but the Network Card is designed to use the CTC in a different mode. On the Local Card, the anticipated use of the CTC is in the timer mode, where the system clock is prescaled and is used to decrement the down counter. However, on the Network Card, the CTC operates in the counter mode. In this mode, the CLK/TRG signal is used to decrement the down counter, which provides the Zero Count/Time Out signal which is used as a clock signal for the SIO. This mode is programmed by outputting the address which develops the CTC $\overline{CE}$ signal and selects the proper channel. Then the required data is placed on the data bus and written to the channel control register to select counter mode and its specific options. After this step, an 8-bit time constant can be loaded into the Time Constant register to be used as a divisor of the signal on the CLK/TRG input. The device does not use the CTC to generate interrupts; therefore, no interrupt vector is loaded nor are interrupts enabled.

The programming of the SIO is described in reference 18 and will only be summarized here. The design of the

Network Card provides that the SIO interact with the Y processor through a series of programmed interrupts. Before the Network Card can become operational, it must be initialized. The initialization process involves performing I/O writes to the eight write registers in each channel. The first write to an SIO channel will cause the data bus to be written into Write Register 0. Writes to this register, include a 3-bit pointer to the location of the next read or write. For example, if the lower 3-bits of the data word were 101, then the next write to that channel would cause data to be written into Write Register 5. Besides the addressing function, Write Register 0 generates controls often used. Write Register 1 enables various interrupts and allows programming of the W/RDY signal. Write Register 2 contains the interrupt vector. If the "status affects vector" feature is programmed, then bits D3, D2 and D1 might be changed subject to the type of interrupt being generated. Write Register 3 sets receive parameters and the number of bits/character. Write Register 4 sets the parity, stop bits, sync function and the clock prescaler for sampling the data. Write Register 5 sets the transmit parameters and the number of bits being transmitted per character. Also bit 7 of this register controls the $\overline{DTR}$ output, $\overline{DTR}$ will be the complement of this bit. Write Registers 6 and 7 define the SYNC characters used in synchronous mode, while just register 6 is used to define the flag character when in SDLC mode.

To complement the Write Registers, there are Read Registers to provide status information on each channel. First, there is Read Register 2 in Channel B only. This register reflects the current status of Write Register, also only in Channel B, which contains the interrupt vector. Read Register 0 provides the status of the receiver and transmitter. Read Register 1 contains information relating to the end of transmit or receive operation. The status bits in this register sometimes only apply in certain modes of operation. After the initialization process is completed, the transmitter will become active upon receipt of the $\overline{CTS}$ signal, while the receiver will become active after receipt of the $\overline{DCD}$ signal. This concludes the discussion of the SIO operation.

Summary

This chapter reviewed the circuitry and operations of the Network Card. The CTC provides the source of timing signals, while the SIO receives, transmits, and processes the data. The remainder of the circuitry on the Card supports buffering and control of the buses and port address decoding. All of these functions could be accomplished by other hardware. However, the configuration of the Card would be changed completely. The SIO is the basis for the design of the Network Card, so that the Network Card can be considered to be an SIO and the supporting circuitry. Other Data Link Control Chips do exist (Ref 5), but their

features vary widely. Therefore, if the Network Card were designed around one of these other chips, the supporting circuitry would have to be modified accordingly.

V. Universal Network Interface Device

This chapter provides an overview of the prototype Universal Network Interface Device (UNID). This prototype was constructed for the purpose of testing the feasibility of a flexible network interface device. As such, the device meets the requirements for testing and for making modifications rather easily. Even so, the design and construction of the device is far removed from a fieldable, production version. With this point made clear, the contents of this chapter will be outlined.

The first section describes the overall architecture of the device. The second section addresses the procedures which must be accomplished prior to using the device as a message processor. This section deals with both the hardware and software procedures required. The third section reviews several specific features relating to the implementation of the device. These features include the generation of the clock signals, an addressing PROM used on the shared memory card, the mini-disk interface and the wiring used in the device.

Overall Architecture

The device consists of two microcomputer boards which share a 16K block of dynamic RAM memory. Accesses to the shared memory are arbitrated by the Dual Processor Card, which also processes the refresh signals generated by each

Z-80 Central Processing Unit (CPU), to provide the necessary refresh signals to the shared memory. The microcomputer boards were designated the X and Y processors. The X processor would interface with one or more Local Cards and the Y processor was used to interface with network traffic through a Network Card. The design of the Local Card includes four RS-232C I/O ports plus on-card programmable timing sources. The Network Card provides two RS-422/423 channels supported by on-card programmable timing sources.

The prototype device uses these cards in a Zilog ZCC-9 Card Cage. This card cage has nine card-slot chassis which provides the type of connectors needed to mount the Zilog Z-80 series cards. These connectors are numbered J1 - J9. The card cage also has wire-wrap pins which can be interfaced with 26-pin ribbon cable connectors. These connectors were used to interface RS-232C compatible terminals directly to the microcomputer boards and as the interface point for control signals to be applied to the device. These 26-pin connectors are numbered J14 - J21. Table I provides a list of the cards and devices which can be interfaced through each of these connectors. Each of the cards and plugs used with these connectors are labeled with the connector number to minimize confusion when constructing the device. Card-slot J5 indicates a Mini-Disk interface. This interface is discussed in the last section of this chapter. Connector J20 is used for

TABLE I

UNID CONNECTOR INTERFACES

Connector Number	Card or Device
J1	Dual Processor Card
J2	Y-Processor Card
J3	X-Processor Card
J4	RAM Memory Board
J5	Mini-Disk Interface
J6	-
J7	Network Card
J8	-
J9	Local Card
J14	-
J15	-
J16	Y-Processor RS-232C Interface
J17	-
J18	X-Processor RS-232C Interface
J19	-
J20	Device Controls Interface
J21	-

device controls. At the present time, these controls consist of debounced switches which provide the $\overline{\text{RESET}}$ signal for each microcomputer board.

The device as configured does not have its own power supply. Power in the form of +5V, +12V and -12V is supplied by an exterior power supply. The device draws about 5 amperes from the 5V supply and less than .1 of an amp from the +12V and -12V supplies. This concludes the discussion on the overall architecture of the device. The next section discusses the procedures necessary to use the device as a message processor.

Procedures Required Before Use

This section provides a description of the procedures necessary before using the device as a message processor. These procedures relate to the types of hardware interfaces used and to the type of software required to support the message processing function. The hardware procedures will be addressed first, followed by the software procedures.

Hardware Procedures. The procedures involved in supporting the message processing function basically involve the numbers and types of interfaces required. Depending on the node configuration, which the device is supporting, there could be many hardware variations in the device. If more than one Local Card and one Network Card were required, then the appropriate jumpering was

required to insure that the I/O address spaces did not conflict. Also, if more than one Local Card was used, the wired-in interrupt vector addresses would have to be modified so that the interrupting USART was being serviced by the correct routine.

The other problem requiring hardware modifications would be the configuration of the device being serviced by a given port. The Local Card shall be considered first. It can support four RS-232C compatible users, presuming that only primary channels are used. However, the user might require full duplex service over the primary and secondary channel, this would require two I/O ports. Thene there is the question of whether the UNID is appearing as Data Communications Equipment or as Data Terminal Equipment. Depending on which configuration is required will define how the signals at each port will be wired to the 25-pin RS-232C female connector. The Local Card is intended to be used to support asynchronous data transfer with lower data rate users; however, the 2651 Programmable Communications Interface (USART) can support synchronous communications. If this feature is required, it will require the addition of one or possibly two additional signals. These signals are Transmitter Signal Element Timing (DCE)/(DTE) and Receiver Signal Element Timing (DCE). Whether the UNID is acting as DCE or as DTE, will determine if the device will be the source and/or destination for these signals.

The Network Card would require modifications similar to those required on the Local Card. The service support required by the port and the CTE/DCE configuration defines the types of jumpering required to support each device. The drivers used on the Network Card are Motorola MC3487's, while the receivers are Motorola MC3486's. This driver/receiver pair is designed specifically to support the balanced and unbalanced voltages associated with the RS-422 and RS-423 Standards, respectively. Since the Network Card is primarily to support synchronous communications, the Transmitter Signal Element Timing and Receive Signal Element Timing signals must be accomodated. The only other hardware consideration deals with loading the software operating system. In a production version of the device PROM's or ROM's would provide the method for software loading. For a given type of service a different type of software would be loaded. The next subsection treats the software aspect in greater detail.

Software Procedures. This subsection addresses procedures required for software support. These procedures are covered in the following order: loading, operating system, initializing, proposed load map and interprocessor communications. As mentioned in the hardware procedures description, the operating system would most likely be loaded by some type of read-only memory. It would not be cost-effective to provide pre-loaded operating

systems for all possible device configurations, therefore, some method is required to modify or initialize the operating system.

The initialization procedure would be handled by as short as possible section of code. A terminal would have to be connected directly with each of the microcomputer boards. After the device is powered up, it would be reset and the initialization process could begin. This process would deal with specifics such as number of terminals, I/O addresses, vectored interrupt addresses, baud rates, numbers of stop bits, sync characters and any other parameters needed to support the operating system and the associated communication mode. This loading and initialization procedure could be implemented as described; however, an alternative method would be to include a terminal and small mass storage device as part of the UNID. This alternative would cause a significant increase in the device's cost, but it would probably be more than worth it in terms of flexibility and having a means of performing diagnostics and monitoring message traffic. Either alternative would support an operating system, but the latter one would probably be superior in the long run.

The device in production version would allow the addition of memory for each processor or to increase the size of the shared memory block. Suppose that the device is configured, so that, each processor has 32K of memory and the shared block also has 32K bytes. Then, a typical

load map of the device might appear as shown in Figure 5-1. A large block of the dedicated memory for each processor would hold the operating system. For both processors, memory would be required to hold data tables and as a scratch memory to perform operations on messages. The largest part of the shared memory would contain message data being held for transmission to the Local Card or to the Network Card. There would also be a portion of the shared memory allocated to such tables as the Local Message Table, the Network Message Table, and the Memory Management Table. These tables would be accessed by each processor to provide a means of controlling message flow and message parameter flow between the processors. A description of how accesses to these tables would be arbitrated is described below.

Consider the Local Message Table. This table would deal with messages going to the Local Card. It would be used to pass parameters such as starting address, stopping address, local device address, and operations to be performed on the message. These parameters plus two software flags would be used to control access to each entry. During the initialization process, one of the software flags would be set to deny access to that entry for one of the processors, while the second flag would be set to allow the second processor to access to that entry. In the case of the Local Message Table, the X processor would be denied access to all entries in the table, while the

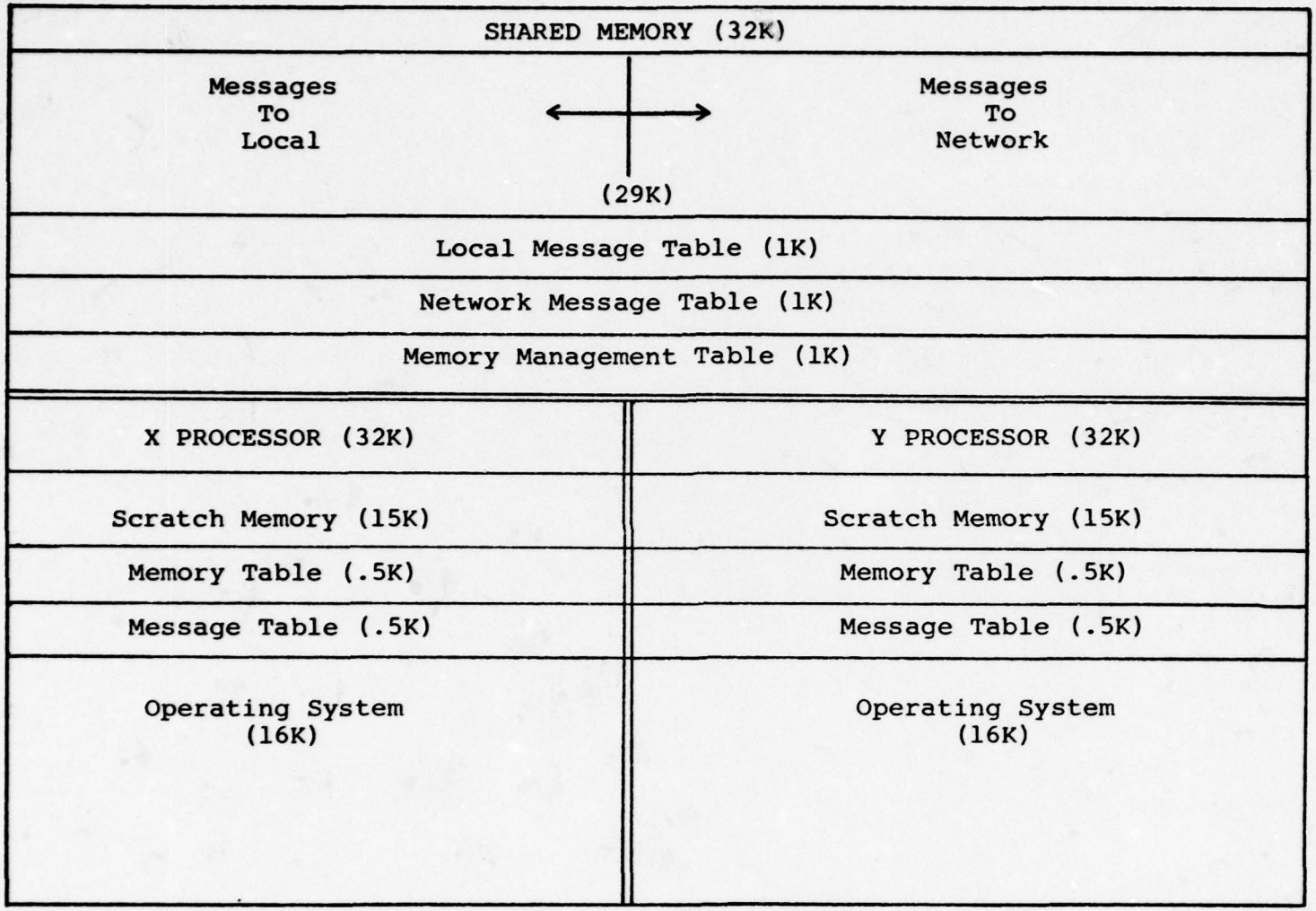


Fig. 5-1. Typical Load Map

Y processor would be allowed access to all entries in the table. Each operating system would contain software, which would poll each entry in the table. The X processor would be searching for an entry relating to a message to a local user. Its software flag would be checked for each entry. The first entry which contains an "access" flag for the X processor would then be fully processed. After the X processor had completed reading the data in the entry, it would then go in and set its software flag to "no access" and set the Y processor software flag to "access". The next time the Y processor needed an entry location to write message parameters, this entry could be accessed by it. When the Y processor completed its data read, it would reset its own flag to "no access" and then reset the X processor flag to "access". This mode of operation would preclude simultaneous accesses to the same data byte and insure data in a given table entry is not changed during an entry read operation. This concludes the discussion on the software procedures. The next section will deal with several features of the UNID implementation.

Implementation of the UNID

This section deals with several aspects of the implementation of the UNID which were not specifically addressed elsewhere in this thesis. These features are the generation of the 180° out-of-phase clock signals, the addressing PROM used on the shared memory card, the mini-disk interface and the wiring used in the device. These features

will be covered in the order listed in the following subsections.

Out-of-Phase Clock Signal. For the memory arbitrator to function correctly, the clocks of the two processors must be 180° out-of-phase. The obvious way of accomplishing this is to invert the clock signal of one processor, and provide it to the second processor. This procedure was accomplished using the X processor clock as the source. The Z-80 Microcomputer Board (MCB) provides an inverted version of the clock signal at pin 99 on the board's edge connector. This signal was wired to pin 39 on the Y processor edge connector by means of the backplane wiring. Pin 39 is a user defineable pin on the Z-80 MCB. Pins 11 and 12 on Integrated Circuit A38 (74161: Synchronous Binary Counter) were cut to disable the Y processor clock. Then, a wire was soldered to connect pin 39 to solder joint A38-12. This connection provided the clock signal to the Y processor. The cutting of the two pins on the counter, also removed the CLK/2 signal. This signal was replaced by inverting the CLK/2 signal from the X processor and applying it to the Y processor CLK/2 pin on the motherboard. These changes in the clock signal effectively disable the Y processor board when it is not used in the J2 slot of the UNID motherboard. The next subsection deals with another modification to one of the Z-80 series boards, that modification involved the replacement of an addressing PROM on the RAM Memory Board.

Addressing PROM. This subsection deals with the replacement of a 32 x 8 bit PROM which is used by the Zilog RAM Memory Board (Ref 19). This PROM is a Harris 7603 or equivalent, which uses the four highest order bits of the address bus along with the $\overline{MREQ}$ and $\overline{RFSH}$ signals to generate the enabling signals for memory accesses and memory refresh. The PROM provided with the Memory Board placed the 16K of memory available into the 1st, 2nd, 3rd, and 8th pages of the address space of the processor. This memory placement conflicted with the memory placement of the RAM on the Micro-Computer Boards. The 4K of RAM on each of these boards had to remain in Page 1 due to software constraints in the system monitor. This conflict caused the addressing PROM to be replaced. No facility was available to program this particular type of PROM so that, it was decided to use simple combinational logic to develop the required signals. The PROM usually provides five signals used by the board. Four of these signals are used as bank select signals for the four banks of memory on the card. The fifth signal is used as a second enabling signal for the RAM on the card. The 16K of memory would occupy one bank of memory on the card so that one of the bank select signals, called Row Address Strobe ($\overline{RAS0}$), and the master RAM enable had to be generated. The combinational logic used to correctly generate the signal is shown in Figure 5-2. The D5 signal corresponds to the $\overline{RAS0}$ signal while the D4 signal acts as the master RAM enable. Table II provides the truth table

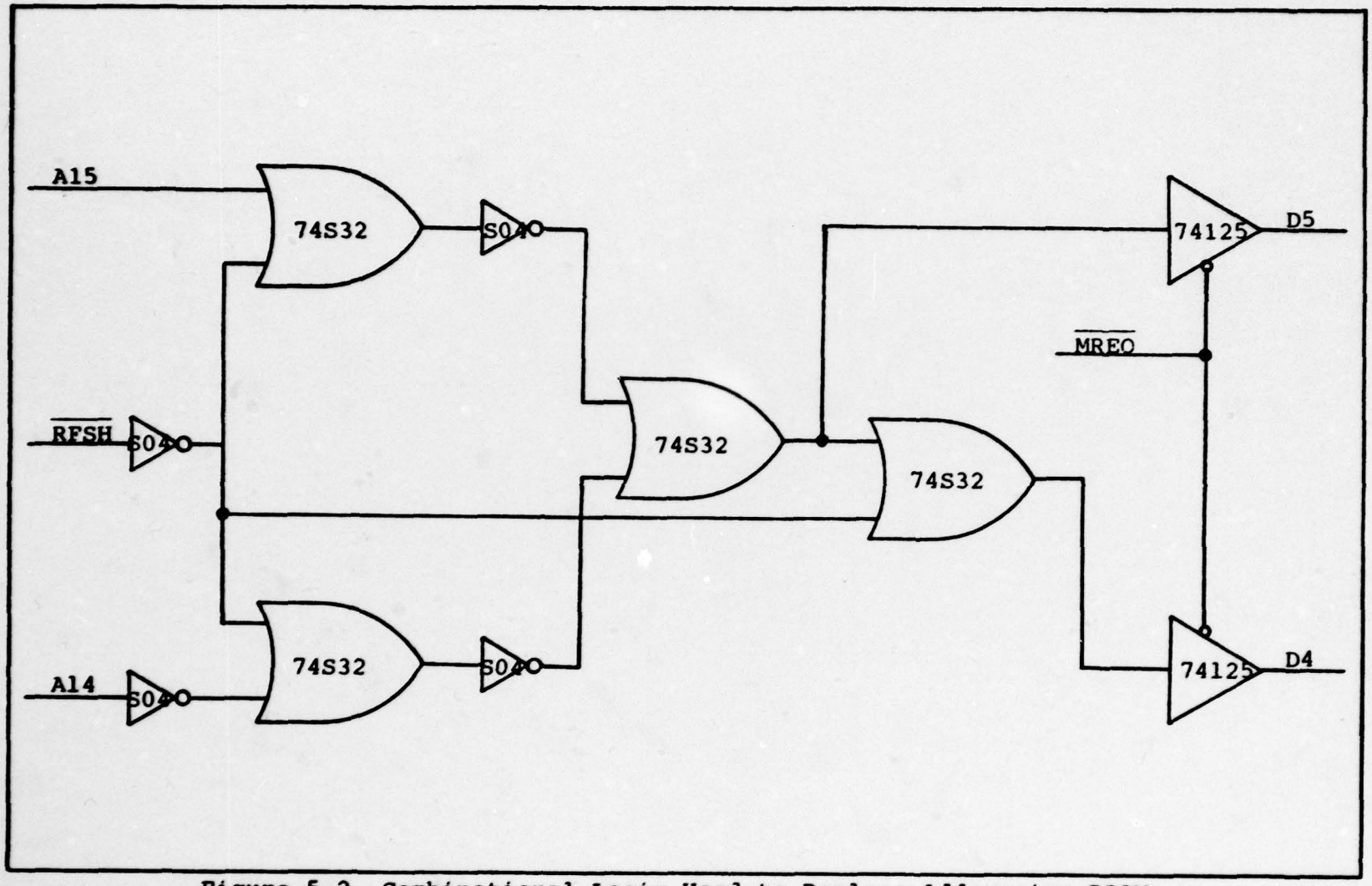


Figure 5-2. Combinational Logic Used to Replace Addressing PROM

TABLE II

COMBINATIONAL LOGIC TRUTH TABLE

$\overline{RFSH}$	A15	A14	$\overline{MREQ}$	D5	D4
X	X	X	H	H	H
L	X	X	L	L	H
H	H	L	L	L	L
H	H	H	L	H	H
H	L	L	L	H	H
H	L	H	L	H	H

for the signals used. A ribbon cable with 16-pin DIP plugs was used to connect the off-board combinational logic to the PROM socket. The logic design places the 16K of RAM into the 8000_H to $BFFF_H$ address space. This completes the addressing PROM discussion, the next subsection will describe the mini-disk interface.

Mini-Disk Interface. The prototype UNID provides card-slot J5 in the motherboard as a point from which to interface the X processor with a North Star Mini-Disk System. At the start of the thesis effort, this system was intended to support software development activities on the UNID. The interface is currently hardware compatible, except that the initializing PROM of the Mini-Disk System requires memory to be available to write the Disk Operating System

into Page 2 of memory. If this PROM can be changed, the system could be used as originally intended. Documentation of the Mini-Disk System interface exists in reference 8, which is available in the AFIT Digital Engineering Laboratory Technical Reference Files. The last subsection of this chapter deals with wiring performed on the UNID.

UNID Wiring. This subsection describes several aspects of wiring of the UNID. The wiring of the device was performed using 30 gauge wire-wrap wire, except for a few places, where 28 gauge wire was used. The type of signal being used defined the color of the wire used. Table III provides the color codes used. These codes were used on

TABLE III

COLOR CODING OF WIRING

COLOR	SIGNAL TYPES
ORANGE	Address Bus
GREEN	Data Bus
YELLOW	CPU Signal or inverse
WHITE	Logic generated signals
BLUE	IEI, IEO, and clock signals
BROWN	I/O to and from Device
VIOLET	+12V, -12V, Jumpering for MCB's
RED	+5V
BLACK	GND

all wires, except on the signal lines from the SIO on the Network Card to its drivers. In this case orange wiring was used in leiu of brown. A wire listing for the back-plane is included in Appendix D. This concludes the discussion on the wiring of the device.

Summary

This chapter was intended primarily to review several of the peculiar aspects in the implementation of the UNID. The discussion is provided to support any efforts on the further development of this device. The sections in this chapter should aid in the understanding of the device, and provide some ideas relating to the direction of further efforts.

There are certain specific areas, which should be considered in this regard. In the prototype version, Zilog Z-80 series boards are used. The MCB's provide for many capabilities (Ref 23), which would not be necessary in a production UNID. The RAM Memory Board could be integrated with the Dual Processor Card. Also once the design was fixed for the Local and Network Cards, it would be desirable to replace the wire-wrap boards with printed circuit boards.

VI. Summary and Conclusions

This chapter provides a brief description of the thesis and provides recommendations relating to follow-on thesis efforts, changes in the device design, and possible projects relating to the device.

Overview of Thesis

This report provides a thorough description of the hardware of the prototype Universal Network Interface Device. The nature of the investigation was such, that the greatest amount of effort was directed to understanding and implementing previous efforts. Therefore, this effort did not result in the development of any significant contributions to the enhancement of man's knowledge. However, it is hoped that what was accomplished can be readily understood and further development continued without much delay. Therefore, the chapters on the three cards and the UNID integration are provided to review what was accomplished. The next section will address follow-on topics and a specific plan relating to further effort on the system.

Follow-On Efforts

There are two areas which would provide topics for follow-on investigations. The first area relates to the software aspects of the device. The software support required to support the UNID in many different applications

is tremendous. This results from the desired flexibility of the device and the fact, that the design requires that the software be the prime source of the flexibility. There would be two aspects to this topic. The first would be the development of the support software general architecture. This would define the methods for creating the different types of operating systems to be used, to perform major changes to the software and to perform minor changes. This architecture would provide the software flexibility without creating an unmanageable software base. The second aspect would be a demonstration operating system, which had changes incorporated within the constraints of the above defined architecture. One area of study would be the trade off between the use of the assembly language or a higher order language.

A hardware investigation would involve extensive testing of the device, implementation of Z-80A processor, expansion of the card cage, incorporation of an internal power supply, implementation of hardware modifications through DIP switches rather than jumpering, and construction of a chassis to support the device and its interfaces. These aspects will be discussed briefly in the paragraphs below.

The testing of the device to date was aimed at the Dual Processor Card with the two other cards receiving little attention. At the present time, the Dual Processor Card does not provide perfect arbitration of the memory accesses.

Testing on the Card shows that there are times that data bytes are incorrectly written to the stored memory, when the other processor is making memory accesses. Therefore testing should be performed relating to the timing of the write signal to isolate the source of this problem. It was found that the detailed drawings and a logic analyzer were indispensible during testing of the device. Minor testing was performed on the Local Card and the Network Card. The address decoding logic and data bus controlling logic tested correctly for both cards. The CTC's on both cards were successfully tested. However, further testing is required to insure that interrupts are handled correctly and the communcations devices can be programmed. It is recommended that these testing activities be confined to the various parts of each board so as to verify the correct operation of all parts of each card, before testing each card as a unit.

Implementation of Z-80A processors is a requirement for the device to meet its original design configuration. This implementation could well cause problems in the arbitration of memory accesses, plus causing the replacement of the microcomputer boards and the RAM Memory Boards.

The last four tasks would not be difficult but they could be time consuming. The DIP switches would deal with at least six I/O channels for a basic device, plus two I/O port address areas and the interrupt addresses on the Local Card. A new card cage is required to support any

configuration more complex than the basic device. The device chassis would be challenging in trying to get an arrangement that would allow for testing and modification, yet minimizing size. The culmination of these six tasks would be a device in a pre-production configuration.

Changes in Device Design

The first change recommended is that the shared memory be replaced with static RAM. The current use of dynamic RAM triples the complexity of the Dual Processor Card. And it would be possible to lose data in memory if enough of the appropriate refresh signals are inhibited. These reasons suggest that Static RAM be used as the shared memory component. The second recommendation deals with the Network Card. The design goal data rate is 1.5 Megabit/sec. The Z80A-SIO can function at 880 Kilobits/sec in half duplex mode. Achievement of the desired data rate might be possible with a Signetic 2652 Multi-Protocol Communications Controller (MPCC). The MPCC data rate specification is 2 Megabit/sec and it includes all features of the Z80-SIO. It can also interface with an 8-bit or 16-bit data bus. Thus, it would be upward compatible with a 16-bit processor if such a requirement became necessary. The last change recommended was stated in the previous section. That change is that the UNID processors be upgraded to Z-80A's.

Recommended Projects

These project recommendations propose lower-level efforts as possible lab projects for a microprocessor interfacing laboratory. The first proposal is that the UNID be interfaced with a separate Zilog microcomputer system which is available in the Digital Engineering Laboratory. The second proposal is that a lab group perform some of the testing activities which were recommended previously. The scope of this effort could be easily modified by requiring the testing of a particular combination of the three cards. Accomplishment of either of these projects would take the implementation of the UNID one step closer to a pre-production version of a flexible message processor which could be used in different types of computer communications network applications.

Bibliography

1. 1842 EEG/EEIC TR 78-5. An Engineering Assessment Toward Economic, Feasible and Responsive Base-Level Communications through the 1980's. Richard-Gebaur AFB, Missouri: 1842 Electrical Engineering Group, 31 OCT 1977.

2. Brown E.F., D.A. Myers and W.A. Riski. "EE 6.54 Final Report". Course Report. Wright-Patterson AFB, Ohio: Air Force Institute of Technology, March 15, 1979.

3. EIA Standard RS-232C. Interface Between Data Terminal Equipment and Data Communication Equipment Employing Serial Binary Data Interchange. Washington, D.C.: Electronic Industrial Association, August 1969.

4. EIA Standard RS-422. Electrical Characteristics of Balanced Voltage Digital Interface Circuits. Washington, D.C.: Electronic Industries Association, April 1975.

5. EIA Standard RS-423. Electrical Characteristics of Unbalanced Voltage Digital Interface Circuits. Washington, D.C.: Electronic Industries Association, April 1975.

6. Intel, Corp. MCS-80 User's Manual. Santa Clara, California: Intel Corporation, October, 1977.

7. Kane, Jerry. An Introduction to Microcomputers Volume 3 Some Real Support Devices. Berkeley, California: Adam Osborne and Associates, Inc., September 1978.

8. Loewer, Bob. "The Z-80 In Parallel," Byte, 3:60-63, 174-176 (July 1978).

9. Motorola Semiconductors Products, Inc. Master Selection Guide and Catalog. Phoenix, Arizona: Motorola, Inc., 1977.

10. Osborne, Adam. An Introduction to Microcomputers Volume 1 Basic Concepts. Berkely, California: Adam Osborne and Associates, Inc., 1976.

11. Osborne, Adam. An Introduction to Microcomputers Volume 2 Some Real Products. Berkeley, California: Adam Osborne and Associates, Inc., September 1978.

12. Sluzevich, Sam C. "Preliminary Design of a Universal Network Interface Device", Master's Thesis. Wright-Patterson AFB, Ohio: Air Force Institue of Technology, December 1978.

13. Texas Instruments, Inc. The Line Driver and Line Receiver Data Book. Dallas, Texas: Texas Instruments, Inc., 1977.

14. Texas Instrument, Inc. The TTL Data Book (Second Edition). Dallas, Texas: Texas Instruments, Inc., 1976.

15. Weissberger, Alan J. Data Communication Handbook. Sunnyvale, California: Signetics, October 1977.

16. Weissberger, Alan J. "Data-Link Control Chips: Bringing Order to Data Protocols", Electronics, Vol. 51 No. 12 (June 8, 1978):104-112.

17. Zilog, Inc. Product Specification Z80-CPU/Z80A-CPU Cupertino, California: Zilog, Inc., March 1978.

18. Zilog, Inc. Product Specification Z80-SIO. Cupertino, California: Zilog, Inc., March 1978.

19. Zilog, Inc. RMB (RMB/E) Hardware User's Manual. Cupertino, California: Zilog, Inc., June 1978.

20. Zilog, Inc. The Z80 Family Program Interrupt Structure. Cupertino, California: Zilog, Inc., October, 1977.

21. Zilog, Inc. Z80 Assembly Language Programming Manual. Cupertino, California: Zilog, Inc., January 1978.

22. Zilog, Inc. Z80-CPU/Z80A-CPU Technical Manual. Cupertino, California: Zilog, Inc., 1977.

23. Zilog, Inc. Z80-MCB Hardware User's Manual. Cupertino, California: Zilog, Inc., January 1978.

Vita

Eric Frederick Brown was born on 20 August 1951 at Ramey AFB, Puerto Rico. He graduated from Beavercreek High School, Beavercreek, Ohio, in 1969 and was accepted to the U.S. Air Force Academy. Upon graduation in June 1973, he received a commission in the USAF and the Degree of Bachelor of Science in Electrical Engineering. He was assigned as a Division Electrical Engineer to the Air Force Weapons Laboratory, Kirtland AFB, New Mexico. From there he was assigned as a Staff Electronics Engineer at Hq SAC, Offutt AFB, Nebraska. He entered the School of Engineering, Air Force Institute of Technology, in June 1978.

Permanent address: 428 Bridgetown Ct.
Satellite Beach, Florida 32937

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFIT/GE/EE/79-8	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Prototype Universal Network Interface Device		5. TYPE OF REPORT & PERIOD COVERED MS Thesis 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) ERIC F. BROWN Capt USAF		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Air Force Institute of Technology (AFIT/EN) Wright-Patterson AFB, Ohio 45433		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS		12. REPORT DATE December 1979 13. NUMBER OF PAGES 77
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

Approved for public release; IAW AFR 190-17
Joeseph P. Hipps, Major, USAF
Director of Public Affairs

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Data Processing
Message Processor
Networks
Protocols

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The thesis describes a prototype version of a flexible message processor designed for computer communications network applications. This message processor is microcomputer based and is called a Universal Network Interface Device. The thesis describes the operational features and construction details of the three cards, which were developed in the construction of the prototype device. These three cards are used with two

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

microcomputer boards and a memory board to provide the capability to interface local users to a computer communications network. The device also provides the capability to act as the node connecting two networks operating under dissimilar protocols.