MICROPROCESSOR INTERFACING TECHNIQUES

SECOND EDITION

AUSTIN LESEA
RODNAY ZAKS

MICROPROCESSOR INTERFACING TECHNIQUES

AUSTIN LESEA

RODNAY ZAKS

U.S.A.
2020 Milvia Street
Berkeley, California 94704
Tel:(415) 848-8233
Telex: 336311

EUROPE
313 rue Lecourbe
75015 - Paris, France
Tel: (1) 8282502
Telex: 200858

SECOND EDITION

ACKNOWLEDGEMENTS

The following persons or companies have supplied valuable information, photographs, programs or schematics of their products or projects: Although not all information supplied could be used, their cooperation is gratefully acknowledged.

Intel, Motorola, Persci (disks), Shugart (disks), Thomson-CSF (CRTC board), David Reinagel (Music Synthesis), Rockwell, Data I/O (programmer), Prolog (programmer), Zilog, Hewlett-Packard (analyzers), Control Data (disks), North Star, Imsai, Altair (S100 bus), Tarbell (cassette interface), Component Sales (keyboards), MOS Technology, Advanced Micro Devices, Fairchild, NEC, Western Digital, Dynabyte (RAM), National Semiconductor, Analog Devices, Lawrence Laboratory, University of California at Berkeley, Power-One (power supply), Fluke, Biomation (analyzer), Trendar(fault analysis)

FOREWORD

Every effort has been made to supply complete and accurate information. However, Sybex assumes no responsibility for its use; nor any infringements of patents or other rights of third parties which would result. No license is granted by the equipment manufacturers under any patent or patent rights. Manufacturers reserve the right to change circuitry at any time without notice.

In particular, technical characteristics and prices are subject to rapid change. Comparisons and evaluations are presented for their educational value and for guidance principles. The reader is referred to the manufacturer's data for exact specifications.

Library of Congress Card Number: 78-55237
ISBN Number: 0-89588-003-2
Printed in the United States of America
Printing 10 9 8 7 6 5 4 3 2 1

CONTENTS

TABLE OF CONTENTS

PREFACE

Computer interfacing has traditionally been an art, the art to design and implement the required control electronics for connecting a variety of peripherals to the main processor.

With the advent of microprocessors, and of LSI chips, since 1976, microprocessor interfacing is no longer an art. It is a set of techniques, and in some cases just a set of components. This book presents the techniques and components required to assemble a complete system, from a basic central processing unit, to a system equipped with all usual peripherals, from keyboard to floppy-disk.

Chapters two and three are a recommended reading for every designer who has not had the experience of designing a basic system. Chapter two presents the construction of a basic CPU, in the case of popular microprocessors such as the Intel 8080, 8085, and the Motorola 6800. Chapter three presents the set of input-output techniques used to communicate with the external world, and a brief survey of the existing chips which facilitate the implementation of these techniques.

Chapter four is an essential chapter: the microprocessor-based CPU will be successively interfaced to every major peripheral: keyboard, LED, teletype, floppy-disk, CRT display, tape-cassette.

The following chapters then focus on specific interfacing problems and techniques, from industrial design (analog-to-digital conversion) (chapter five) to communication with the outside world (busing, including S-100 and other bus standards), in chapter six.

Chapter seven presents a detailed case study, which incorporates the interfacing principles presented in the previous chapters: the design of a real 32-channel multiplexer.

Finally, chapter eight presents the basic techniques and tools for trouble-shooting microprocessor systems.

This book assumes a basic understanding of microprocessor systems, equivalent to the level of book **C201 - Microprocessors: from chips to systems.**

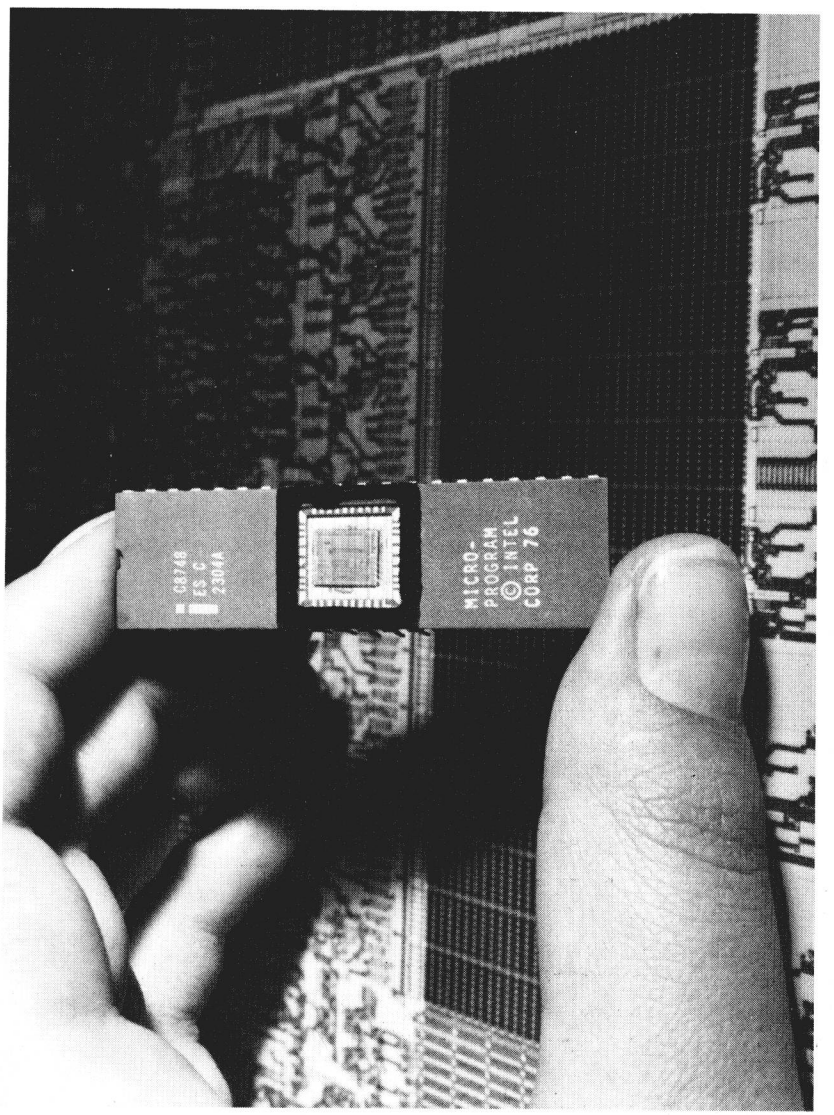

Fig. 1-0: Intel 8748

CHAPTER 1
INTRODUCTION

OBJECTIVE

The objective of this book is to present the complete set of techniques required to interface a microprocessor to the external world. Because of the availability of new LSI interface chips, which implement most techniques in hardware, it will be shown that interfacing has become simple.

FROM ART TO TECHNIQUE

Microcomputer interfacing has traditionally been the art of designing complex boards of logic managing the data transfers and the synchronization signals necessary for the processor to communicate with external devices. The processor itself has traditionally required one or more boards of logic. Each I/O interface has traditionally required one or more boards of logic. Such Multi-board implementations are obsolete today in most cases. *Large scale integration* (LSI) has now resulted in the implementation of a complete (or almost complete) CPU in *a single chip*. The new market created by microprocessors has introduced, in turn, the necessity for manufacturers to provide the required support components. Most of the boards required to assemble a complete system have now been shrunk into LSI chips. Since 1976, even device-controller interface chips exist. They do for interface design what the microprocessor has done for CPU design.

A complete interface board, or most of it, is today shrunk into a few LSI chips. The price paid, just like in the case of a microprocessor, is that the architecture is frozen inside the LSI chip.

It is now possible to implement a complete microcomputer system, including interfaces, in a small number of LSI chips. *If you are still implementing your interfaces on one or more boards of logic, your design might be obsolete!*

Microprocessor interface-chips have not reached their maturity yet. They are still "dumb" chips. In other words, they can execute only a very few commands. It can be predicted that in view of the very low cost of a processing-element, most microprocessor interface chips will become fully programmable in the near future. They will become "processor-equipped", and be capable of sophisticated programmed sequencing. They will become "intelligent" interfaces.

Although this next step has not been reached yet, all the techniques presented within this book should retain their validity in the future. There is always a trade-off between software and hardware implementation. The balance will change with the introduction of new components, and with the trade-offs involved in each specific system design.

THE HARDWARE/SOFTWARE TRADE-OFF

Detailed techniques will be presented to solve all the common interfacing problems. As usual in computer design, most of these techniques may be implemented either by *hardware* (by components), or by *software* (by programs), or by a combination of both. It is always up to the system designer to strike a reasonable compromise between the efficiency of hardware, and the lower component count of a software implementation. Examples of both will be provided.

THE STANDARD MICROPROCESSOR SYSTEM

Throughout this book, reference will be made to a "standard microprocessor." The "standard" microprocessor today is the *8-bit microprocessor*. Examples are the Intel 8080, 8085, the Zilog Z-80, the Motorola 6800, the Signetics 2650, etc. In view of the pin number limitation on DIP's (dual-in-line packages), the 8-bit microprocessor has become the norm. The reason is simple:

The number of pins is limited to 40 (or 42) by economic considerations. Industrial testers required to test components having more than 40 pins are either not available, or would be extremely expensive. All standard testers will accept only up to 40 or 42 pins. In addition, naturally, the cost of the package itself increases rapidly over 40 pins.

Because of the limitation of the densities which can be achieved with the MOS LSI process, it is not yet possible to integrate the complete memory, plus I/O facilities directly on the microprocessor chip. In the standard system, the microprocessor itself (abbreviated MPU), and perhaps the clock, reside on a single chip. The memory (ROM, or Read-Only Memory, and RAM, or Random-Access Memory) are external. Because memory and I/O chips are external to the microprocessor, a selection mechanism must be provided to address the components: a microprocessor must be equipped with an *address-bus*. The standard width of the address-bus is 16 bits, permitting the addressing of 64 K locations (where K = 1,024: 2^{16} = 64K).

An 8-bit microprocessor will transfer 8-bit data. It must be equipped with an 8-bit *data-bus*. This requires 8 additional pins.

At least two pins must be provided for power, and two more for connection to an external crystal or oscillator. Finally 10 to 12 control lines must be provided to provide the coordination of data transfers in the system (the control-bus). The total number of pins used is 40. No pins are left unused.

Because of this pin-number limitation, a 16-bit microprocessor cannot provide at the same time a 16-bit address-bus, and a 16-bit data-bus. One of the buses must be *multiplexed*. This results in turn into a slower operation, and in the necessity of external components to multiplex and de-multiplex the buses.

It can be expected that the progress of integration will soon introduce a new standard microprocessor, the *16-bit microcomputer-on-a-chip*. A microcomputer-on-a-chip is a microprocessor-plus-clock-plus-memory (ROM + RAM) on a single chip. Since the memory is directly on the chip, there is no longer the necessity to provide an external address-bus. 16 pins become available. In such a system, *at least 24 lines become available for data transfers*. They are general-purpose I/O lines. The disadvantage of current microcomputers is that, for the time being, the quantity of memory which may be implemented directly on the microcomputer-chip is limited. The current limitation is 2048 words for the ROM, and 512 words for the RAM. Adding external memory involves complex multiplexing and de-multiplexing, and is usually not worth it. However, if a system can be implemented in the near future with a significantly larger memory, it can be expected that it will become the next standard design.

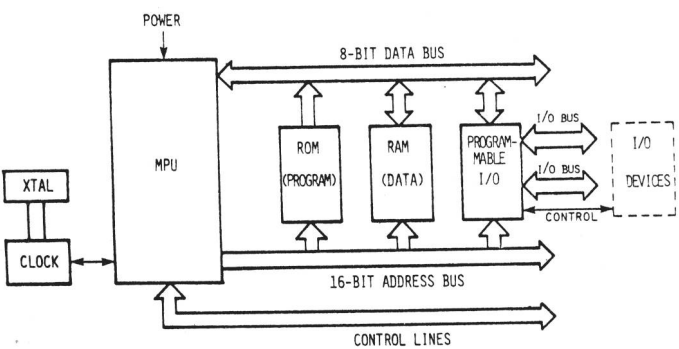

Fig. 1-1: Standard Microprocessor System

For the time being, the 8-bit microprocessor is indeed the standard design used for "powerful" and flexible applications, and will be referenced as

such. The basic diagram showing the architecture of a standard system appears on Fig. 1-1. The microprocessor itself, labeled MPU, appears on the left of the illustration. On most standard systems, up to 1976, the clock was external to the MPU. It appears here on the far left at the illustration. Since 1976, the clock circuitry has been incorporated in the microprocessor chip itself and all recent products do not require this external clock. However they always require an external *crystal* or oscillator. It appears here, connected to the clock.

The microprocessor creates *three buses:*

The 8-bit bi-directional *data-bus* (implemented in tri-state logic to allow the use of a direct-memory-access controller, or DMAC).

A 16-bit mono-directional *address-bus,* connected internally, within the microprocessor, to the address-pointers, and in particular to the program-counter (PC). The address-bus is also implemented in tri-state logic in order to allow the use of a DMAC.

Finally, a 10 to 12-line *control-bus*, which carries the various synchronization signals to and from the microprocessor. Control lines are not necessarily tri-state.

All the usual system components are directly connected to these three buses. The three basic components appear on the illustration. They are respectively the ROM, the RAM, and the PIO. The *ROM* is the Read-Only Memory. It stores the *programs*. The *RAM* is the Random-Access-Memory. It is a read-write MOS memory which stores the *data*. The PIO is a programmable input-output chip which multiplexes the data-bus into two or more input-output ports. It will be studied in more detail in chapter three. These ports may be connected directly to input-output devices, or to device-controllers, or may require the use of interface-circuits.

The interface-circuits or *interface-chips* required to interface this basic system to actual I/O devices will be connected to these buses, whether the microprocessor buses or the input-output buses created by the PIO, or by other chips.

Interfacing techniques are precisely those techniques required to connect this basic system to the various input-output devices. The basic interfacing techniques required to connect any microprocessor system to input-output devices are essentially identical. They will be described in detail in chapters three, four, and five. At the level of the microprocessor itself, the logical and electrical interface required is simple. All standard microprocessors have essentially the same data-bus and the same address-bus. The essential difference is the *control-bus*. It is the specific characteristics of the control-bus which make input-output interface chips compatible or incompatible from one microprocessor to the next. As an example of basic interfacing characteristics, the basic 8080, 6800, and 6502 SC/MP, interfacing characteristics appear on Fig. 1-2.

Interfacing input-output devices requires the understanding of two basic techniques:

1. The assembly of a complete CPU, using a microprocessor chip. This topic will be addressed in chapter 2.
2. The fundamental input-output techniques used to communicate between the microprocessor and the external world. This topic will be addressed in chapter 3.

MICROPROCESSOR CONTROL SIGNALS

It has been shown that a standard MPU creates three busses: the 8-bit bi-directional data bus, the 16-bit mono-directional address bus, and a control-bus of varying width, depending on the microprocessor. The data-bus is essentially identical for all microprocessors. It is 8-bit-bi-directional bus, normally implemented in tri-state logic. Similarly, the address-bus is almost universally a 16, or sometimes 15-bit mono-directional bus, used to select a device external to the MPU. The actual use and interconnect of the address-bus and the data-bus will be presented in the next chapter. The third bus is the only complex one. It carries the microprocessor control signals or "interface signals."

The control bus provides four functions:

1. memory synchronization
2. input-output synchronization
3. MPU scheduling -- interrupt and DMA
4. utilities, such as clock and reset.

Memory and input-output synchronization are essentially analogous. A hand-shake procedure is used. In a "read" operation, a "ready" status or signal will indicate the availability of data. Data will then be transferred on the data-bus. In the case of some input-output devices, an "acknowledge" is generated, to confirm the receipt of data. For "write" operation, the availability of the external device is verified through a status-bit or signal, and the data is then deposited on the data-bus. Here also an "acknowledge" might be generated by the device to confirm the receipt of data.

The generation, or non-generation, of an "acknowledge" is typical of the use of the synchronous procedure versus an asynchronous one. In a synchronous procedure, all events take place within a specified period of time. In this case there is no need to acknowledge. In an asyn-

APPROXIMATE SIGNAL EQUIVALENCES

	8080&8228	8085	Z-80	6800	6502
ADDRESS	A0–A15	AD0-AD7 +ALE A8-A15	A0-A15	A0-A15	AB0-AB15
DATA	D0-D7	AD0-AD7 +–ALE	D0-D7	D0-D7	DB0-DB7
CONTROL	HLDA	HLDA	BUSAK	BA&VMA	----------
	HOLD	HOLD	BUSRQ	HALT	RDY
	ø2	CLK	--	ø2 stretched	ø2 stretch
	INT	INTR	INT	IRQ	IRQ
	INTE	--	--	--	--
	WAIT	--	--	--	--
	READY	READY	WAIT	--	RDY
	RESET	RESET	RESET	RESET	RESET
	SYNC	--	M1	--	SYNC
	INTA	INTA	M1&IORQ	VMA&FFF8	--
	MEMR	RD&IO/M	RD&MEMRQ	R/W&ø2	R/W&ø2
	MEMW	WR&IO/M	WR&MEMRQ	as above	as above
	I/O RD	RD&IO/M-	RD&IORQ	as above	as above
	I/O WR	WR&IO/M-	WR&IORQ	as above	as above
	BUSEN	--	--	HALT	--
	SSTB	--	--	--	--
OTHER CONTROL SIGNALS	--	RST 5.5	--	--	--
	--	RST6.5	--	--	--
	--	RST 7.5	--	--	--
	--	TRAP	NMI	NMI	NMI
	--	RESET	--	--	--
		OUT			
	--	SID	--	--	--
	--	SOD	--	--	--
	--	ALE	--	--	--
	--	--	RFSH	--	--
	--	--	HALT	--	--
	--	--	--	TSC	--
	--	--	--	DBE	--
	--	--	--	--	SO

Fig. 1-2: Signal Equivalences

chronous system, an acknowledge must be generated. The choice of a synchronous versus an asynchronous communication philosophy is basic to the design of a control bus. A synchronous design has a potential for a higher speed and a lower number of control lines. However it imposes speed constraints on the external devices. An asynchronous design will require an additional acknowledge, and somewhat more logic, but allows the use of components of varying speeds in the same system.

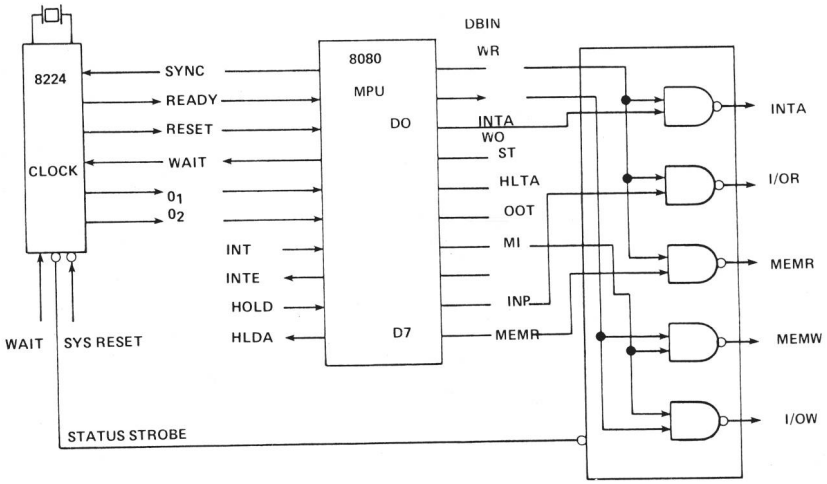

Fig. 1-3: 8080 Control Signals

As examples the 8080 Control signals are illustrated in Fig. 1-3 with the bus timing in Figures 1-4 and 1-5. In contrast, the 6800 bus is shown in Figures 1-6 and 1-7. In chapter two these buses will be explained. Chapter six expands on buses and describes some standard buses in use today.

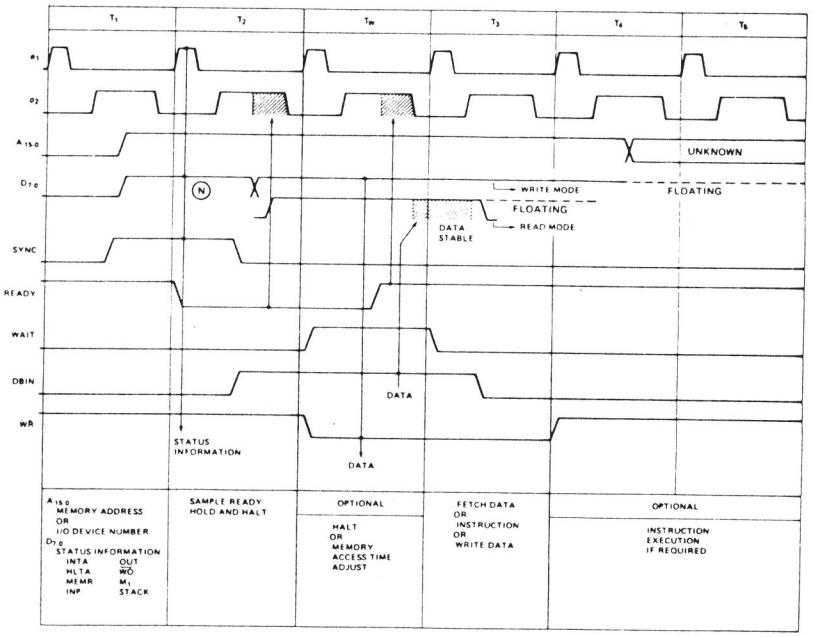

Fig. 1-4: Basic 8080 Instruction Cycle

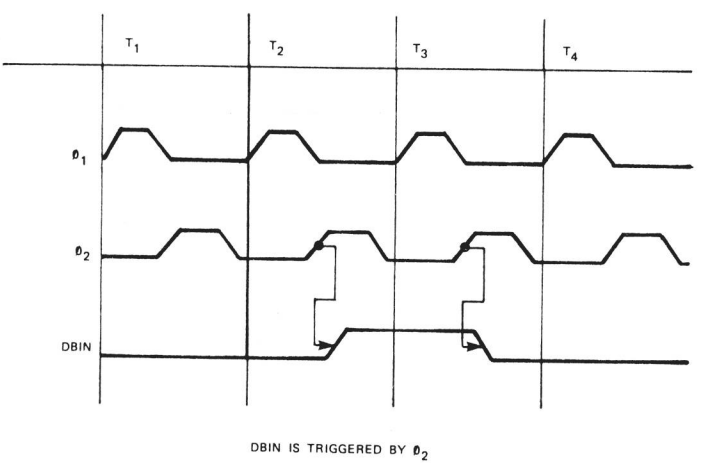

Fig. 1-5: DBIN Timing

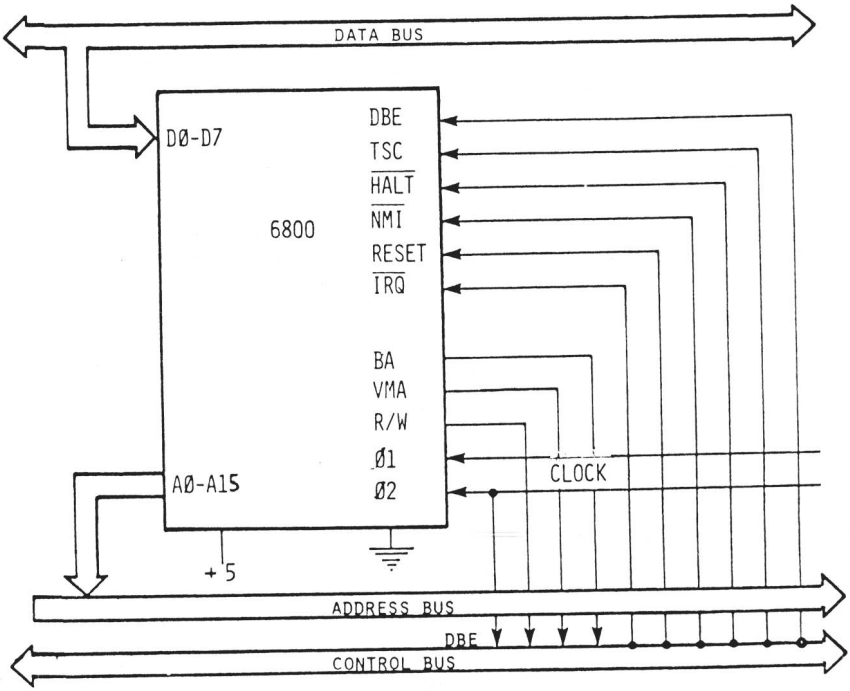

Fig. 1-6: 6800 Bus Signals

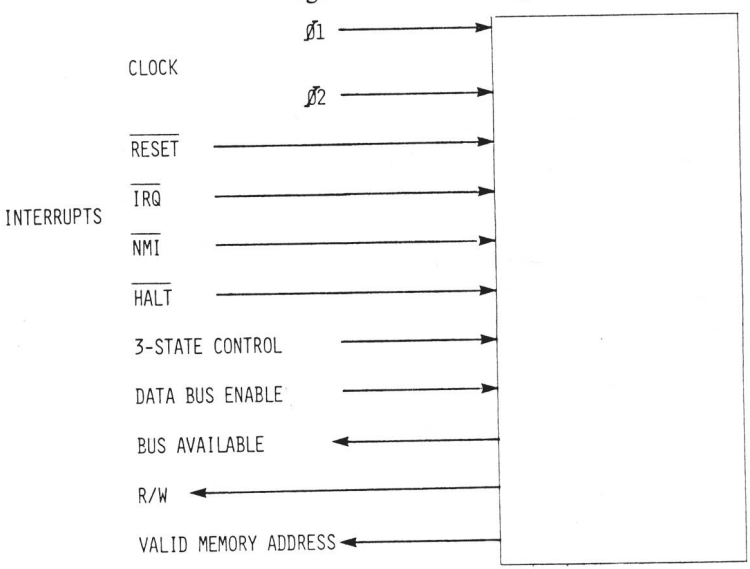

Fig. 1-7: Detail: 6800 Bus Control

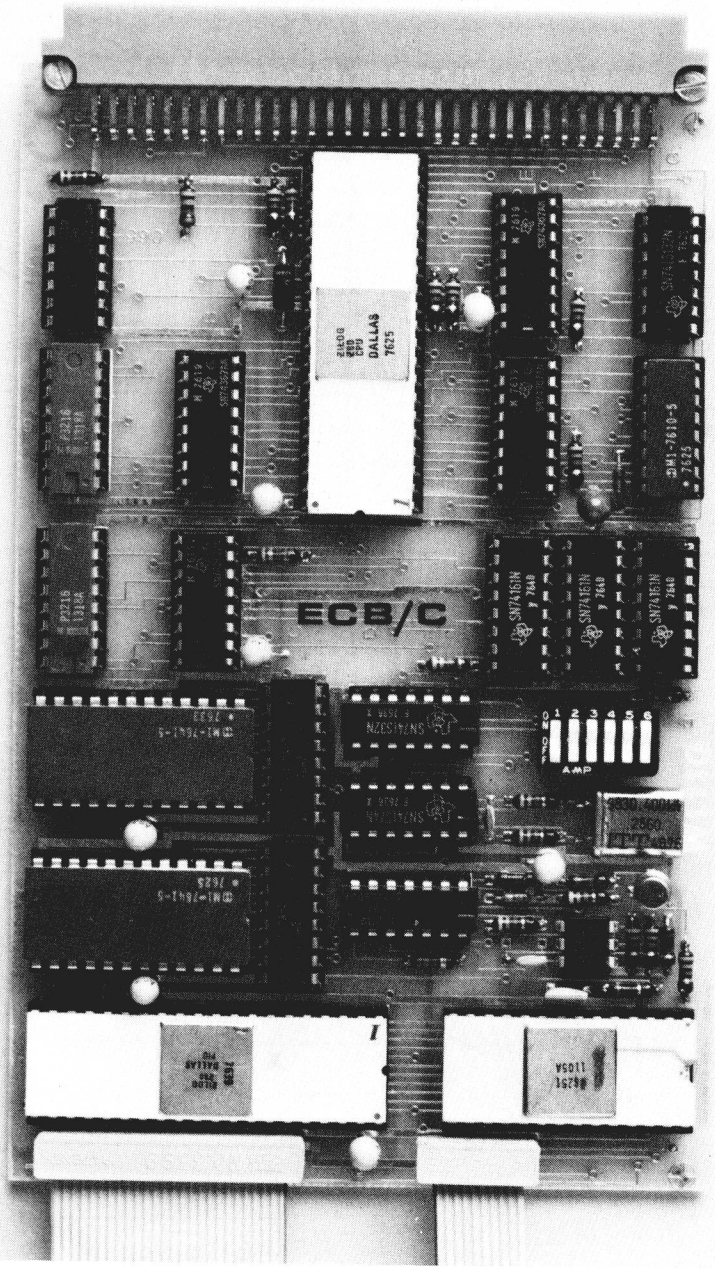

Fig. 2-0: Z-80 CPU Board