RICHARD C. HALLGREN

interface projects for the apple II

$12.95 A SPECTRUM BOOK

RICHARD C. HALLGREN

INTERFACE PROJECTS FOR THE APPLE II

A SPECTRUM BOOK

Prentice-Hall, Inc., Englewood Cliffs, New Jersey 07632

Library of Congress Cataloging in Publication Data

Hallgren, Richard C.
Interface projects for the Apple II.

"A Spectrum Book."
Includes index.
1. Computer interfaces. 2. Apple II (Computer)
I. Title.
TK7887.5.H34 621.3819'592 82-3748
ISBN 0-13-469395-7 AACR2
ISBN 0-13-469387-6 (pbk.)

This Spectrum Book is available to business and organizations at a special discount when ordered in large quantities. For information, contact Prentice-Hall, Inc., General Publishing Division, Special Sales, Englewood Cliffs, N.J. 07632.

A SPECTRUM BOOK. Printed in the United States of America.

10 9 8 7 6 5 4 3 2

Editorial/production supervision by Kimberly Mazur
Manufacturing buyer: Barbara A. Frick

ISBN 0-13-469395-7

ISBN 0-13-469387-6 {PBK.}

Prentice-Hall International, Inc., *London*
Prentice-Hall of Australia Pty. Limited, *Sydney*
Prentice-Hall Canada Inc., *Toronto*
Prentice-Hall of India Private Limited, *New Delhi*
Prentice-Hall of Japan, Inc., *Tokyo*
Prentice-Hall of Southeast Asia Pte. Ltd., *Singapore*
Whitehall Books Limited, *Wellington, New Zealand*

Richard C. Hallgren, Ph.D., is an assistant professor at Michigan State University. He has written numerous articles and books, including INTERFACE PROJECTS FOR THE TRS-80, published by Prentice-Hall.

CONTENTS

1

INTRODUCTION

The rapid expansion of the electronics industry and their ability to mass produce reliable, large-scale integrated circuits has led to the reality of scientific calculators and multifunction digital watches for under $20 and "personal computers" starting from $200. In 1971 the Intel Corporation had the honor of initiating this revolution by introducing the first commercially available microprocessor. While the performance of this device was limited by its relatively slow central processing unit and its use of a 4-bit data bus, its acceptance was so dramatic that within a two-year period the cost of a single unit had dropped from $200 to under $20. Today, it is a rare individual who does not have a microprocessor (if not a personal computer) of some type in his or her home.

As I have talked to owners of personal computers, it has become apparent that there is a certain group of individuals who would like to use their machines as extensively as is possible, both at home and at their place of employment. This inevitably results in the need for interface circuitry to allow the computer to be connected to external devices allowing users to monitor and control their environment. On the basis of the response of readers to several articles that I have written, I have come to the conclusion that there is a great need for a book that covers not only the theory behind interfacing a computer to external devices but also gives a broad selection of interface circuits that can be built and expected to work. What I have attempted to do is to provide you, the reader, with a number if examples of interface circuits ranging from the very simple to the somewhat complicated. I have included software for each circuit and can assure you that both the hardware and the software have not only been tested together but have been used in some practical area of application. The result is a document that explains the theory behind computer interfacing and gives you the choice of either building the circuit as is or modifying it to meet your specific needs. The hardware for these projects was designed and the components selected so that construction and check-out would be a straightforward matter. The circuits have been chosen to offer something of interest and application to a broad range of individuals: the hobbyist, the experimenter, the manager of a manufacturing plant, and the engineer or scientist in a research facility. The intent being to enable people to more fully use the computational and control capabilities of their personal computer in a practical and interesting way.

The book assumes that readers have a fairly good understanding of the commands in Applesoft BASIC, and have written some of their own programs. Since some of the supporting software will be using 6502 machine code, the reader is encouraged to become familiar with the 6502 instruction set. Programing examples using both Applesoft and 6502 machine language have been provided throughout the book. While construction of the circuits is straightforward, it should be understood that a certain amount of experience, and a certain level of expertise is expected. Previous construction of a commercial kit would probably qualify most individuals. So, with all this in mind, let's get started with the job of connecting your Apple II to the outside world.

2

REVIEW OF DATA TRANSFER FORMATS

BASIC INTERFACE CONCEPTS

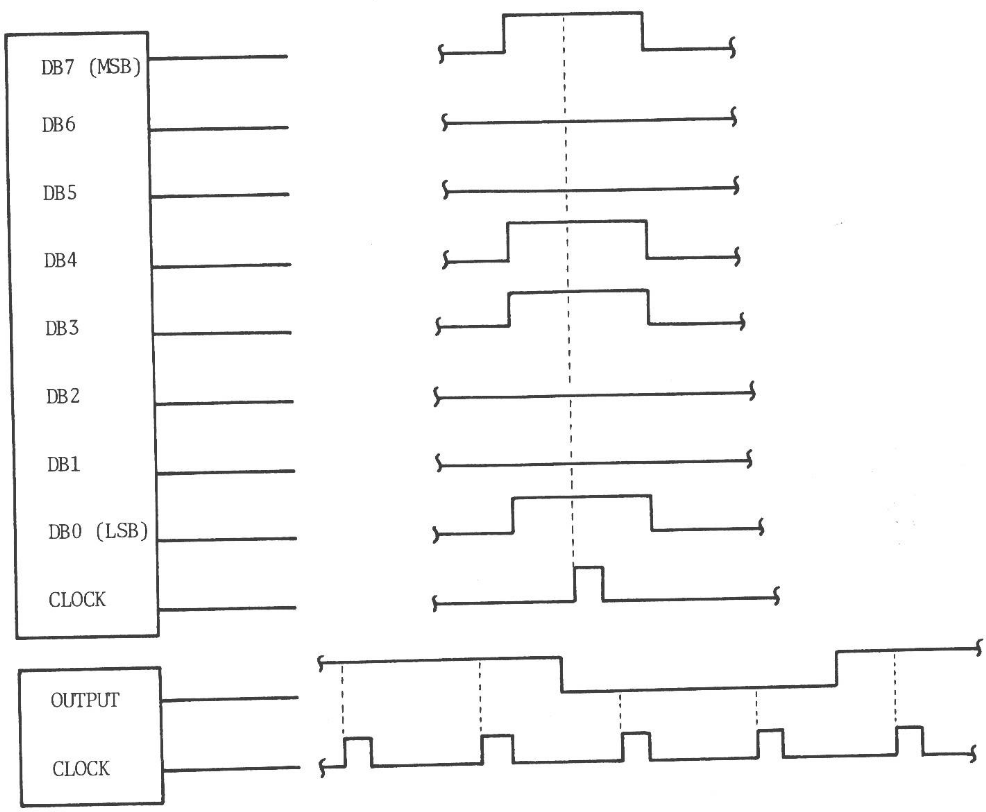
Before we get into actual circuit designs, we need to spend some time reviewing basic interface concepts. Even experienced circuit designers should read this section to become familiar with terminology that will be used.

We can define an input/output (I/O) port as a collection of electronic circuits, under the control of the computer, which routes data to and from an external peripheral device. The key words in this definition are *data* and *external*. The route by which the data flow to or from the external device is called the *port*. A line printer is an example of an external device; the computer sends characters to be printed to the printer and, in some cases, the printer sends control signals back to the computer. The purpose of these signals is to regulate the flow of the data. This interaction is called *handshaking*.

Ports can be constructed so that data are handled in either a serial or a parallel format (Figure 2.1).

Parallel transfer of data involves the mass transfer, at a given point in time, of several bits of data, where the number of bits transferred is usually equal to the word size of the computer. For the Apple II this is equal to 8 bits. In general, the number of bits transferred at one time will be equal to the size of the data bus. For example, if we were working with a Z8000, which has a 16-bit data bus, a parallel transfer would involve 16 bits. When high rates of data transfer are desired, such as data transfer between a computer and a floppy disk, the transfer is usually handled with a parallel format. Computers handle parallel communication rela-

FIGURE 2.1 Comparison of parallel and serial format. The clock impulse is used to signal the external device that the signals are stable. For both examples, the data word being transmitted is binary 10011001.



tively easily and, even though multiconductor cables are required, the total expense is not great because the distance between devices is usually quite short.

Serial transfer of data involves transmitting individual bits of data, one bit at a time. Microcomputers do not normally communicate in a serial format, so there is usually not a single machine language command that facilitates this type of operation. Therefore, we need to add either a software subroutine or a new piece of hardware to accomplish a serial transfer of data. Such an addition results in increased system expense, but gives the capability of transferring data over increased distances at relatively high data rates. The Electronics Industry Association (EIA) RS-232C electrical specification for serial transfer of data is the standard for industrial applications. This specification defines the following voltage levels: A logic level 1 is called a *mark* and is considered to be any voltage level more negative than −3 volts. A logic level 0 is called a *space* and is considered to be any voltage level more positive than +3 volts. In general, designers use +12 and −12 volt levels for the logic 0 and logic 1 states. In addition to specifying voltage levels, the EIA also defines the standard RS-232C connector to be a 25-pin, D subminiature type (commonly referred as a DB-25). The pin assignments and their functions are listed in Table 2.1.

Before we look at specific examples, we should take a look at the architecture of the 6502 to determine how we can communicate in an orderly manner with several peripheral devices that may be connected to the same data bus and to the same address bus. The Apple II computer uses a 6502 type microprocessor to perform the logical, mathematical, and decision-making operations necessary for high-speed operation of the computer. This microprocessor is an 8-bit device which combines the bus structure of the 6501 microprocessor (16-bit address bus, 8-bit bidirectional data bus, and two interrupts) with an instruction set that includes over 150 different commands. The bidirectional data bus is used to transfer information both to and from the central processing unit. The 16-bit address bus has the capability of uniquely defining 65,536 addressable locations. The 6502 generates several control signals which are used both internally and externally to supervise and manage the flow of information. Since data can only flow in one direction at a time, and since the data are usually

TABLE 2.1 EIA standards for DB-25 connector pin assignments when used for communication between RS-232C systems.

Pin 1	PGND - Protective Ground This is chassis or equipment ground. It may also be tied to signal ground.
Pin 2	TD - Transmit Data This is the serial data from the terminal to the remote receiving equipment. When no data is being sent it is in a marking (1) condition.
Pin 3	RD - Receive Data This is the serial data from the remote equipment which is transmitted to the terminal.
Pin 4	RTS - Request to Send Controls the direction of data transmission. In full-duplex operation an "on" sets transmit mode and an "off" sets non-transmit mode. In half-duplex operation an "on" inhibits the receive mode and an "off" enables it.
Pin 5	CTS - Clear to Send Signal from the modem to the terminal indicating ability to transmit data. An "on" is "Ready" and an "off" is "not ready."
Pin 6	DSR - Data Set Ready Signal from the modem to the terminal. An "on" condition indicates that the modem is ready.
Pin 7	SGND - Signal Ground
Pin 8	CD - Carrier Detect An "on" indicates reception of a carrier from the remote data set; "off" indicates no carrier is being received.
Pin 20	DTR - Data Terminal Ready "On" connects the communication equipment to the communications channel; "off" disconnects the communications equipment from the communications channel.
Pin 22	RI - Ring Indicator An "on" indicates that a ringing signal is being received on the communications channel.

directed to a specific device or memory location, these control lines provide an essential coordinating function.

Figure 2.2 shows the pin configuration for the 6502. The 6502 belongs to the one-address architectural class of microprocessors; that is, all memory-related instructions refer to a single memory location. The direction of all data transfers is controlled by the READ/WRITE* (R/W*) line. When this line goes low, all data transfers will occur in the direction from the processor to memory. When it goes high, all data transfers will occur in the direction from memory to the processor. The timing of all data transfers is controlled by the system clock. The clock system used by the 6502 is referred to as *two-phase;* it is most easily thought of as two nonoverlapping square waves. When the microprocessor is reading a data byte from memory, the address of the desired memory location is placed on the address bus during the phase one clock pulse. Information stored in that particular memory location is strobed onto the data bus and into the accumulator

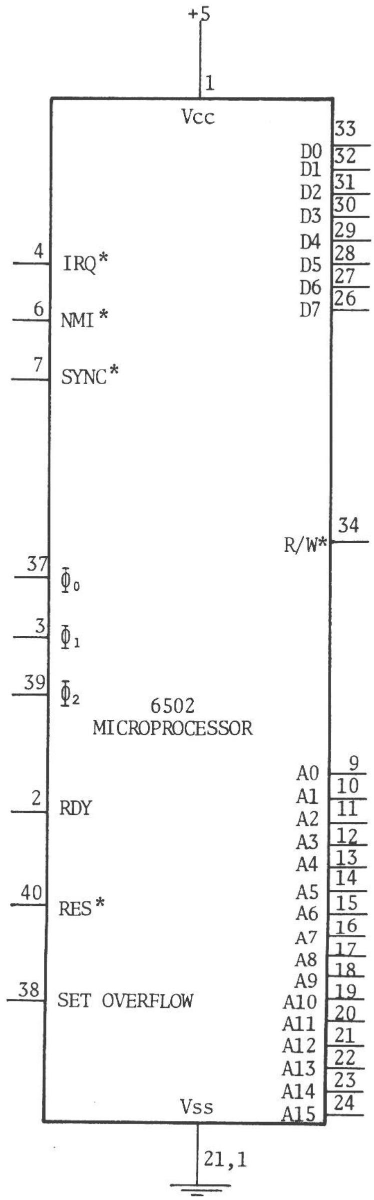


FIGURE 2.2 6502 pin configuration.

during the phase two clock pulse. Data being written to memory is handled by a reversed set of operations.

Fortunately for the individual who wishes to interface the Apple II to external devices, the designer of the computer has provided eight peripheral-board connectors on the mother board. Having these connectors on the mother board allows you to place

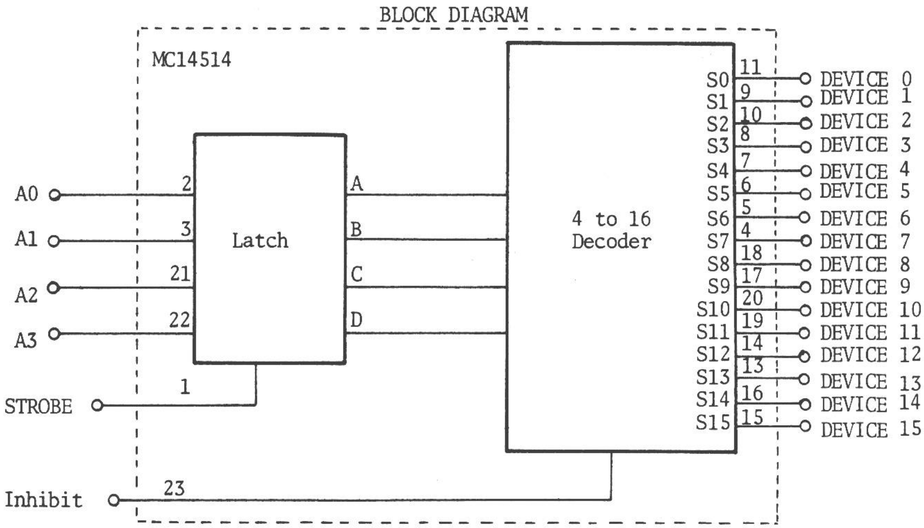
TABLE 2.2 Peripheral I/O location address assignments.

Hex Address	Assigned Function	Comments
C080-C08F	Device Select #0	Pin #41 (DS*) on the selected peripheral connector goes low during phase two of the clock.
C090-C09F	Device Select #1	
C0A0-C0AF	Device Select #2	
C0B0-C0BF	Device Select #3	
C0C0-C0CF	Device Select #4	
C0D0-C0DF	Device Select #5	
C0E0-C0EF	Device Select #6	
C0F0-C0FF	Device Select #7	

any interface boards that you construct inside the computer's case, and permits you to use the Apple's power supply. The peripheral-board connectors give the hardware designer access to all address, data, and control lines. In addition, certain control and address lines have been decoded to provide the designer with a device select (DS*) signal. What this means is that each peripheral connector has a unique range of address locations that will result in a low output on the DS* line. Since DS* goes low during the phase two clock pulse, it is convenient to use this as a data strobe pulse during either a READ or WRITE operation. Table 2.2 shows the address locations that are assigned to each of the eight peripheral connectors.

Since each peripheral connector can address up to sixteen devices, and since we will not exhaust this capability on any one of the projects that you will be building, I will limit my comments here to one peripheral connector, specifically peripheral I/O location 7. In Chapter 5, however, I will design the digital plotter interface so that it resides in I/O location 5. I will do this so that the high speed A/D converter and the digital plotter can be used to digitize and plot data without having to switch the physical location of the circuit boards. Figure 2.3 shows a simple but effective method for decoding the four least significant bits of the address bus into sixteen unique control lines. For normal operation, the INHIBIT control line would remain low. Upon execution of either a READ or WRITE command, the device control signal (DS*), connected to the STROBE control line, would be used to strobe the current contents of the address bus into the decoder. The decoded output would then be used to indicate to a specific external circuit that the data bus was available for its specific use. In the next section we will see how we can use the R/W* control line to direct the flow of data to and from our interface projects.

FIGURE 2.3 4-16 line address decoder. Each device is selected by a unique combination of the address lines A0-A3.



DECODE TRUTH TABLE (Strobe = 1)

INHIBIT	DATA INPUTS A3	A2	A1	A0	SELECTED OUTPUT LOGIC "1"
0	0	0	0	0	Device 0
0	0	0	0	1	Device 1
0	0	0	1	0	Device 2
0	0	0	1	1	Device 3
0	0	1	0	0	Device 4
0	0	1	0	1	Device 5
0	0	1	1	0	Device 6
0	0	1	1	1	Device 7
0	1	0	0	0	Device 8
0	1	0	0	1	Device 9
0	1	0	1	0	Device 10
0	1	0	1	1	Device 11
0	1	1	0	0	Device 12
0	1	1	0	1	Device 13
0	1	1	1	0	Device 14
0	1	1	1	1	Device 15
1	X	X	X	X	All Outputs = 0

X = Don't Care

PARALLEL DATA FORMAT

Transmission and reception of data in a parallel format combines the advantages of high transfer rates and low cost. As mentioned earlier, parallel data transfer involves the mass movement of several bits of data at one time. It becomes a relatively straightforward task to construct both input and output ports since all the control, data, and address lines that we will need are available on the Apple II peripheral I/O connector (see Figure 2.4). Table 2.3 lists the signals appearing on this connector and their functions. Figures 2.5a and 2.5b show examples of practical circuits which could be connected directly to the I/O connector 7, providing one 8-bit input port and one 8-bit latched output port. To transfer data from the accumulator to the latched output port, you need only to perform the BASIC statement POKE-16143,X or the machine language commands LDA #$X and STA $C0F1. The microprocessor performs the following bit transfers when you execute either of these commands:

1. The device address (C0F1) is strobed onto the address bus.
2. The contents of the accumulator (X) are strobed onto the data bus.

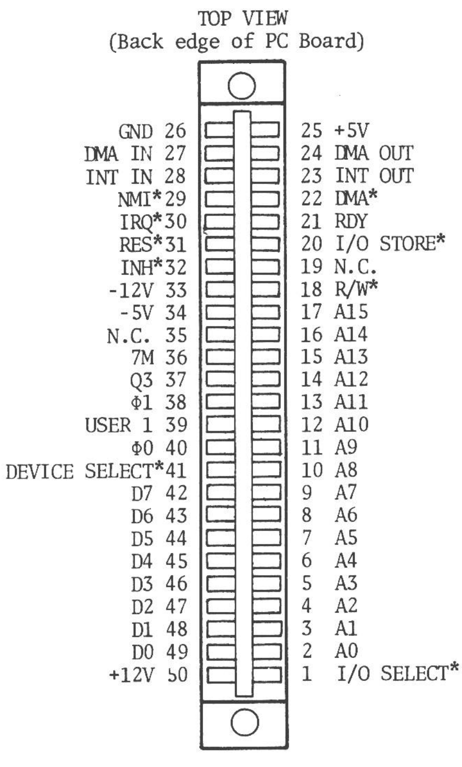


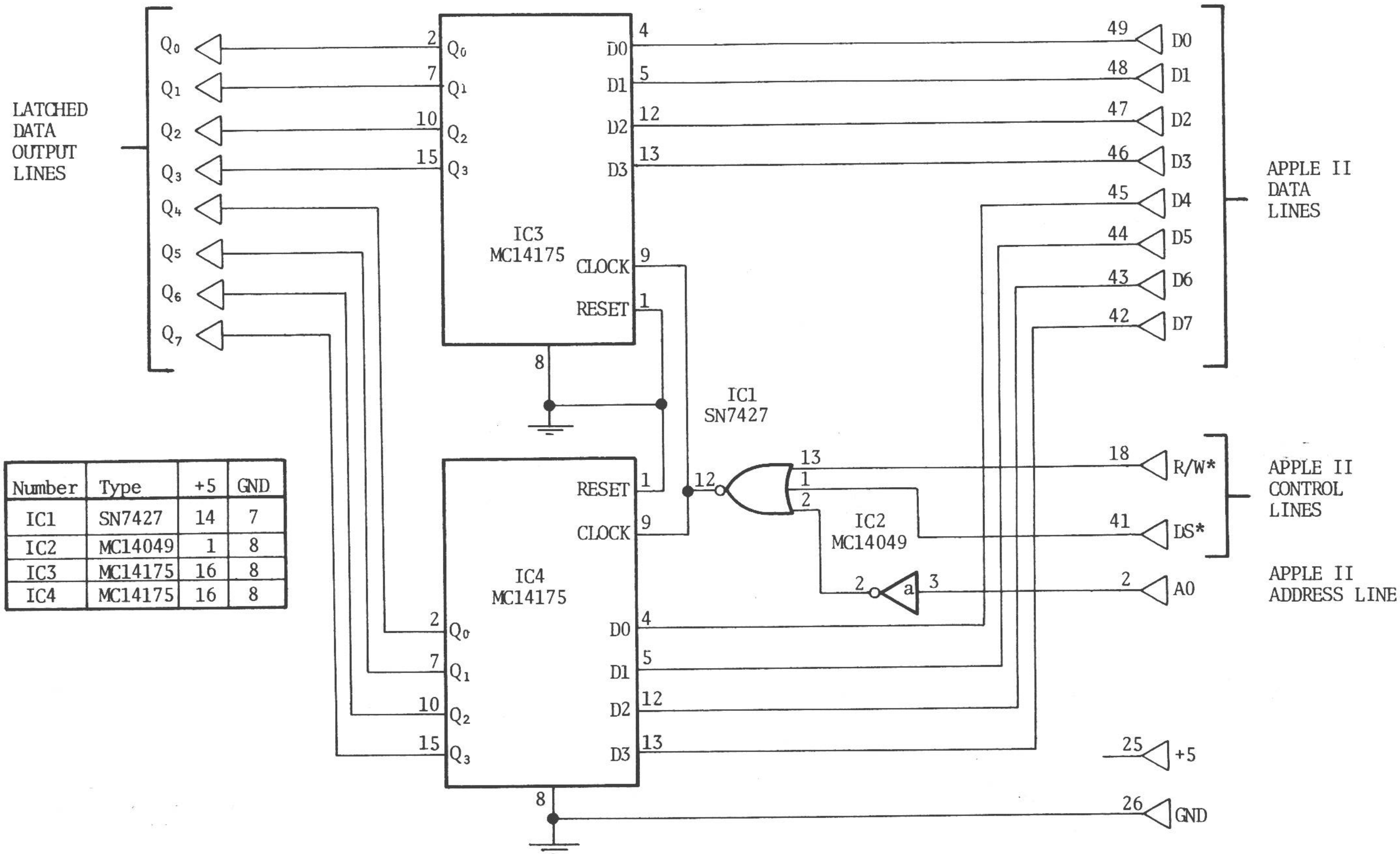
FIGURE 2.4 Output pin configuration for the Apple II peripheral I/O connector. Refer to Table 2.3 for a description of each pin.

TABLE 2.3 Functional description of pins on the Apple II peripheral I/O connector.

P/N	Signal Name	Description
1	I/O SELECT*	Peripheral I/O Select Line
2	A0	Address Output
3	A1	Address Output
4	A2	Address Output
5	A3	Address Output
6	A4	Address Output
7	A5	Address Output
8	A6	Address Output
9	A7	Address Output
10	A8	Address Output
11	A9	Address Output
12	A10	Address Output
13	A11	Address Output
14	A12	Address Output
15	A13	Address Output
16	A14	Address Output
17	A15	Address Output
18	R/W*	System Read/Write Strobe
19	N.C.	No Connection
20	I/O STRBE*	Peripheral I/O Strobe Line
21	RDY	Peripheral Device Ready
22	DMA*	Direct Memory Access Line
23	INT OUT	Interrupt Chain Output
24	DMA OUT	Direct Memory Access Chain Output
25	+5	Positive 5-Volt Supply
26	GND	Power Supply Ground
27	DMA IN	Direct Memory Access Chain Input
28	INT IN	Interrupt Chain Input
29	NMI*	Non Maskable Interrupt
30	IRQ*	Interrupt Request Line
31	RES*	Reset Line
32	INH*	Inhibit Line
33	-12	Negative 12-Volt Supply
34	-5	Negative 5-Volt Supply
35	N.C.	No Connection
36	7M	7 MHZ Clock
37	Q_3	1 MHZ Timing Signal
38	Φ1	Phase 1 Clock Signal
39	USER 1	Disables Internal I/O Adr Decode
40	Φ0	Phase 0 Clock Signal
41	DS*	Device Select Line
42	D7	Bidirectional Data Bus
43	D6	Bidirectional Data Bus
44	D5	Bidirectional Data Bus
45	D4	Bidirectional Data Bus
46	D3	Bidirectional Data Bus
47	D2	Bidirectional Data Bus
48	D1	Bidirectional Data Bus
49	D0	Bidirectional Data Bus
50	+12	Positive 12-Volt Supply

Note: "*" means logical "Ø" true input or output.

Since the data are valid for only a few clock cycles (perhaps 500 ns.), a set of clocked flip-flops (IC3 and IC4) are provided so that the data are latched and consequently stable, until the next command to write data to this port is executed. The circuit works in the following manner:



Number	Type	+5	GND
IC1	SN7427	14	7
IC2	MC14049	1	8
IC3	MC14175	16	8
IC4	MC14175	16	8

FIGURE 2.5a Circuit diagram of a typical parallel output port. Performing a POKE -16143,16 would cause the DS* and R/W* control lines to go low, address line A0 to go high, and 16_{10} to be latched into the output register.