$1.00 ■ OCT. 1976

40-CHANNEL CB
NEW FCC RULES
see page 6

Radio-Electronics®

THE MAGAZINE FOR NEW IDEAS IN ELECTRONICS

punch up your
CB PERFORMANCE
know about SWR

from our lab
2 HI-FI TESTS
of new gear

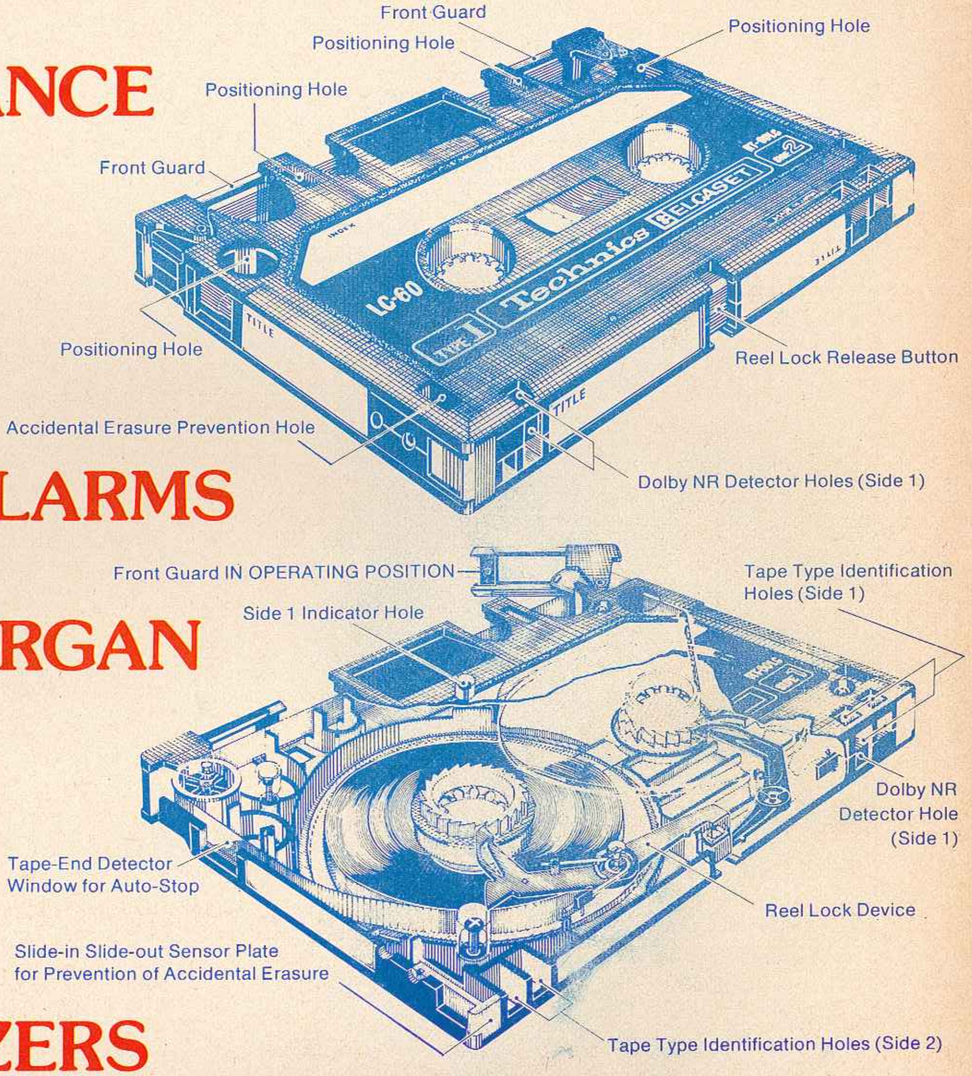
use IC's
OVERSPEED ALARMS

build your own
DIGI-COLOR-ORGAN
add dancing lights

ready now
AM STEREO
broadcasting

how they work
CB SYNTHESIZERS
use fewer crystals

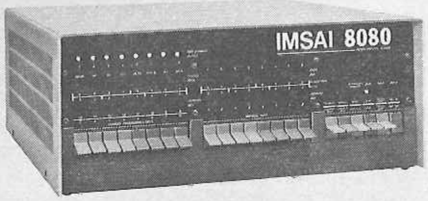
new tape format
"ELCASET"
for hi-fi


Front Guard
Positioning Hole
Positioning Hole
Positioning Hole
Front Guard
Technics
ELCASET
LC-60
TYPE I
INDEX
TITLE
TITLE
Positioning Hole
Reel Lock Release Button
Accidental Erasure Prevention Hole
Dolby NR Detector Holes (Side 1)
Front Guard IN OPERATING POSITION
Tape Type Identification Holes (Side 1)
Side 1 Indicator Hole
Dolby NR Detector Hole (Side 1)
Tape-End Detector Window for Auto-Stop
Reel Lock Device
Slide-in Slide-out Sensor Plate for Prevention of Accidental Erasure
Tape Type Identification Holes (Side 2)

A GERNSBACK PUBLICATION

PLUS

- ★ Komputer Korner
- ★ Brainwave Monitor
- ★ Delayed Sweep Scopes
- ★ Jack Darr's Service Clinic



KOMPUTER KORNER

JONATHAN A. TITUS, PETER R. RONY AND DAVID G. LARSEN*

OOOOPS!

The Komputer Korner that appeared in the July 1976 issue was mistakenly credited to Jonathan A. Titus, Peter R. Rony and David G. Larsen.

We would like to extend our deepest apologies to Tim Barry. He, in fact, actually authored that column.

WHEN DATA IS TRANSMITTED BETWEEN A microcomputer and an input/output device, three actions must simultaneously occur:

1. The microcomputer must select the specific input/output device that will either receive or transmit eight bits of data.
2. The microcomputer must indicate to the specific input/output device when the bidirectional data bus is available for data transmission.
3. The data must be transmitted between the microcomputer and the input/output device in a very short period of time, typically of the order of microseconds.

In preceding columns, we have discussed *accumulator I/O*, in which data is exchanged between the accumulator and an external I/O device. There is a significant disadvantage associated with such an interfacing technique: there exists only a single origin or destination for data. A typical microprocessor chip, such as the Intel 8080, has in addition to the accumulator a variety of internal general-purpose registers that can exchange information with memory. These registers include the B, C, D, H and L registers, each of which is an 8-bit register. From a programming standpoint, it would be very useful to be able to exchange data between any of these registers and any external I/O device. This is the subject of this month's column.

If you desire to exchange data between a general purpose register and an external I/O device, you employ an exciting interfacing technique called *memory I/O* or *memory mapped I/O.* The basic gimmick behind this technique is quite simple: *you treat the input/output device as if it were one or more memory locations.* By doing so, you provide yourself with the opportunity to employ memory instructions in the 8080 microprocessor instruction set. These instructions transfer data between registers and memory locations.

The differences between accumulator I/O and memory I/O can be best understood with the aid of a specific example of an interface between an 8080-based microcomputer and an external I/O device. In this case, the "device" is the Intel 8255 programmable peripheral interface (PPI) integrated circuit. This IC has 24 I/O pins that are shown as PA, PB, and PC in either Fig. 1 or Fig. 2. These pins can be wired directly to any digital device that has TTL-compatible signals.

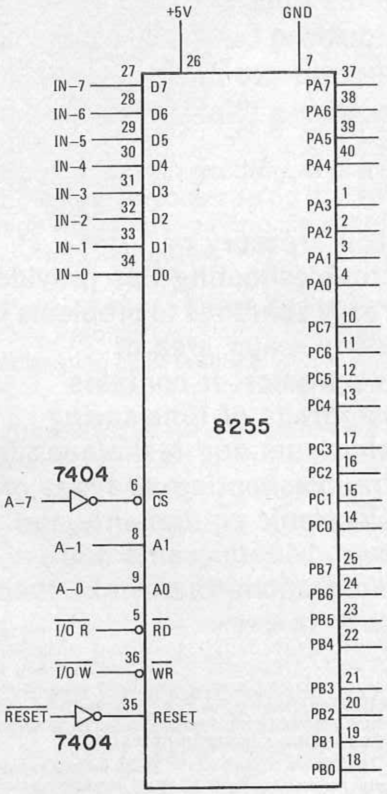


FIG. 1

An 8255 IC appears to an 8-bit microcomputer as either four different external I/O devices or else four different memory locations. Within the IC there are four 8-bit registers that are addressed by the microcomputer:

- Port A—an 8-bit port that can be configured as either an input port, an output port, or a bidirectional I/O port.
- Port B—an 8-bit port that can be configured as either an input or output port.
- Port C—an 8-bit port that can be configured as an input or output port or a pair of control ports—one for port A and the other for port B.
- An internal 8-bit control register that determines the specific I/O

continued on page 24

* Mr. Titus is president of Tychon, Inc., a microcomputer consulting firm in Blacksburg, Virginia. Dr. Rony, Department of Chemical Engineering, and Mr. Larsen, Department of Chemistry, are with the Virginia Polytechnic Institute & State University.

KOMPUTER KORNER

continued from page 22

configuration of the 8255 IC and can be altered at any time by the microcomputer.

As can be seen in Figs. 1 and 2, the 8255 IC contains, in addition to the three I/O ports, an 8-bit bidirectional data bus (D0 through D7) that communicates directly with the 8080 IC; six control inputs–$\overline{CS}$, A1, A0, $\overline{RD}$, $\overline{WR}$, reset, and two power inputs.

Accumulator I/O

In accumulator I/O, you employ the $\overline{I/O\ R}$ and $\overline{I/O\ W}$ function pulses to read from and write into the 8255 chip.[1] The IC is addressed with the aid of bits A0 through A7 (or A8 through A15) on the 16-bit memory address bus of the 8080 microprocessor. In Fig. 1, we use A7 as the IC-select input and A0 and A1 to select one of the four different registers present within the IC. For example, the following program writes information from the 8080 IC into port A.

323_8 Enable the register for port A and allow it to accept data from the accumulator register
200_8 Device code for port A

The port is treated as an I/O device and an accumulator I/O instruction, 323_8, is used. A simple change in the control register will turn port A into an input port and permit the use of the program to read information into the 8080 IC from port A.

333_8 Enable the register for port A and allow it to send data to the accumulator register
200_8 Device code for port A

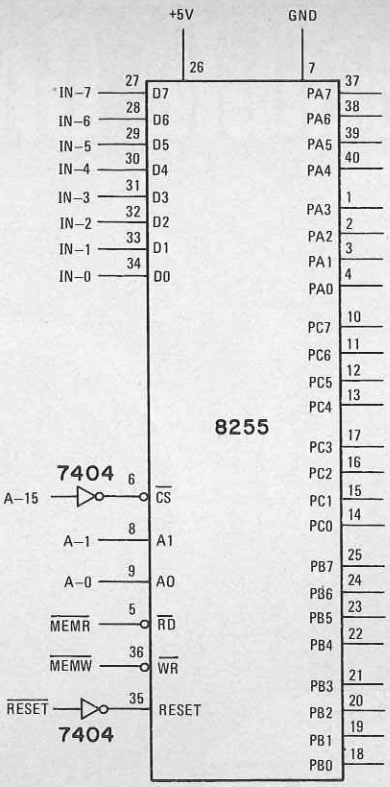


FIG. 2

Memory I/O

In memory I/O, the $\overline{MEMR}$ and $\overline{MEMW}$ memory read/write function pulses are used to exchange data between the internal registers within the 8080 IC and the 8255 registers.[1] The entire 16-bit memory-address bus can be used to address the IC. As shown in Fig. 2, bit A-15 is used as the IC-select input and bits A0 and A1 as the register-select inputs. To output data from register B to port A, the following program can be used:

041_8 Set the 16-bit memory ad-
000_8 dress pointer register with-
200_8 in the 8080 microprocessor chip to the memory address of port A
160_8 Move the contents of register B to port A

Once port A has been selected, you can successively output data from other registers in the 8080 microprocessor:

161_8 Move the contents of register C to port A
162_8 Move the contents of register D to port A
163_8 Move the contents of register E to port A
167_8 Move the contents of the accumulator to port A

Each additional transfer of data requires only 2-ms execution time, which is quite a bit faster than the 5 ms required for successive accumulator I/O data.

We have demonstrated that both the accumulator I/O and the memory I/O techniques are applicable to the 8255 IC. The specific application will determine the best

continued on page 26

KOMPUTER KORNER

continued from page 24

I/O technique. In some cases, accumulator I/O is best; in others, memory I/O simplifies programming and speeds the transfer of large quantities of data to or from memory. It should be noted that some microprocessors only permit memory I/O interfacing techniques. Such IC's frequently possess special memory addressing instructions that speed execution time for memory I/O addressing.

The principal advantage of the 8255 IC is not in programming or execution time, but rather in the ease of wiring of an interface to an external I/O device such as an analog-to-digital converter, a digital-to-analog converter, a digital panel meter, or a digital multimeter. No flip-flops, decoders, or gates are required for the interface; they are all contained within the 8255 IC. In most cases, only SN7404 inverters may be needed to match logic levels between port C, which is usually employed as a control port, and the control pins on the external digital I/O device. It is possible that in the future manufacturers of these semiconductor devices will provide I/O interfaces that will permit a digital instrument to be tied directly to a programmable peripheral interface IC. **R-E**

[1] The accumulator I/O pulse abbreviations, $\overline{\text{I/O R}}$, $\overline{\text{I/O W}}$, $\overline{\text{MEMR}}$, and $\overline{\text{MEMW}}$ are those used by the Intel Corporation, Santa Clara, California. $\overline{\text{I/O R}}$ and $\overline{\text{I/O W}}$ respectively correspond to $\overline{\text{IN}}$ and $\overline{\text{OUT}}$, which we have used in preceding columns.

KOMPUTER KORNER 2

TIM BARRY

NOW THAT WE HAVE HAD SOME ARTICLES INTRODUCING the basic 8080 hardware features, we can begin to study its instruction set and programming characteristics. This is the first of three columns in which we will present some of the basic concepts of all computer instruction sets. These concepts will then be illustrated with the 8080 microprocessor.

The instruction sets of all general purpose computers can be broadly divided into four groups:

1. Data Transfer Instructions
2. Arithmetic/Logic Instructions
3. Control Transfer Instructions
4. Processor Control Instructions

Each of these general groups can then be broken down into sub-groups based upon the architecture of the processor being used. This month we will look at the data transfer instructions and their operation.

Data transfer instructions

Instructions that are used to transfer data from one location in the computer to another without modifying it are all considered to be members of this group. The versatility of a computer's instruction set is directly related to the number of direct data-transfers that can be performed; the more direct data-transfers possible, the easier the computer will be to use.

When analyzing a computer's data transfer instructions, it is useful to think of the entire computer as a collection of *data resources*. A data resource is considered to be any register, memory location, or I/O device. A data resource whose contents can be moved by a data transfer instruction is considered to be a *data source*. A data resource that can receive the data transferred by a data transfer instruction is considered to be a *data destination*. In most computers, any given data resource can be either a data source or a data destination depending upon the instruction being executed. The exception to this occurs with the I/O devices. An I/O device is generally only a data source or a data destination. Thus, an I/O address used for both read and write operations with the same device will probably be communicating with two physically different registers. For example, if you have your teletypewriter hooked up to input port 10 and output port 10, input port 10 would be the data source and output port 10 would be the data destination.

Data transfer notation

When examining a computer's data transfer instructions, we will use a notation designed to illustrate the action performed by the transfers. These notations are defined here and they will be used from now on whenever we need to illustrate a data transfer.

Notation	Meaning
⟶	Data transfer
⟷	Data exchange
( )	Contents of a data source
[Addr]	Memory location addressed by Addr

continued on page 28

KOMPUTER KORNER

continued from page 26

([Addr])	Contents of memory location addressed by Addr

The following are some examples of how the above notations would be used and their meaning:

(A) → B	would denote a transfer of the contents of A into B.
(A) ↔ (B)	would denote an exchange of the contents of A and B.
(A) → [100]	would denote a transfer of the contents of A into memory at location 100.
([100]) → A	would denote a transfer of the contents of memory location 100 into A.

8080 data resources

Data transfer instructions in the 8080 may use any of the following data resources as either data sources or data destinations:

1. An 8-bit register (A, B, C, D, E, H, L)
2. A 16-bit register or register pair (BC, DE, HL, PSW, SP, PC)
3. A memory location addressed by a 16-bit immediate address or a 16-bit register
4. Immediate data
5. An I/O device

In the presentation of the 8080 instructions we will use the following notations to represent these data resources:

Symbol	**Data resource**
S or D	An 8-bit register (A, B, C, D, E, H, or L)
M	The contents of memory as addressed by the HL register pair
RP	One of the 16-bit registers BC, DE, HL, or SP
RS	One of the 16-bit registers BC, DE, HL, or PSW, where PSW is composed of A and the processor flags
RD	One of the 16-bit registers BC or DE
Addr	A 16-bit memory address
IO	An 8-bit I/O address
D8	8-bit immediate data
D16	16-bit immediate data

In addition, some instructions will make reference to specific 8080 registers. These will be pointed out as required. All instructions introduced in the following sections will be presented using the standard mnemonics introduced by Intel Corporation when they first released the 8080. The number in parenthesis next to the instruction is the number of bytes of memory occupied by the instruction. Thus MVI D,D8(2) would be the mnemonic representing the two-byte instruction used to transfer an immediate data value consisting of 8 bits into one of the 8-bit destination registers (A, B, C, D, E, H, or L).

Register to register data transfers

The first group of data transfer instructions we will consider are those that use internal processor registers as both the data source and data destination. The 8080 allows us to transfer the contents of any 8-bit register into any other 8-bit register. In addition, there are a limited number of transfers that use the 16-bit registers. The following are the register-to-register-data transfer instructions listed by their mnemonics (in bold type), the operation performed in transfer notation and the meaning of the instruction.

MOV D,S(1)
Operation performed: (S) → D
The contents of the source register are transferred into the destination register.

XCHG(1)
Operation performed: (HL) ↔ (DE)
The contents of the HL register are exchanged with the contents of the DE register.

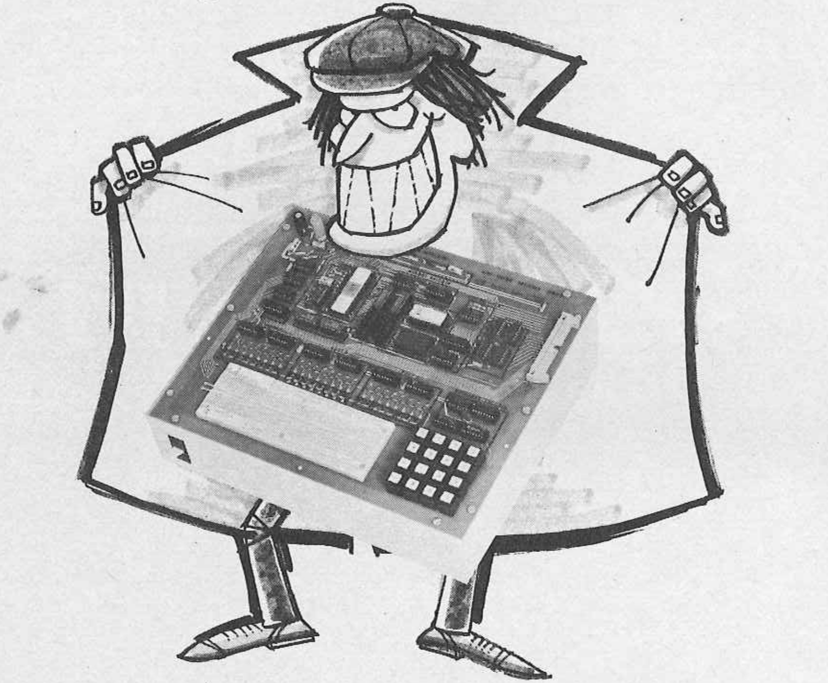
PCHL(1)
Operation performed: (HL) → PC
The contents of the HL register are transferred into the program counter.

SPHL(1)
Operation performed: (HL) → SP
The contents of the HL register are transferred into the stack pointer.

Memory data transfers

This class of instruction uses an internal register or register pair as one data source or

continued on page 30



KOMPUTER KORNER

continued from page 28

destination and a memory location or locations as the other.

LDA Addr(3)
Operation performed: $([Addr]) \longrightarrow A$
The contents of memory as addressed by the 16-bit address included with the instruction are transferred into the A register.

STA Addr(3)
Operation performed: $(A) \longrightarrow [Addr]$
The contents of the A register are transferred into the memory location specified by the 16-bit address included with the instruction.

LHLD Addr(3)
Operation performed: $([Addr]) \longrightarrow L,$
$([Addr+1]) \longrightarrow H$
The contents of memory as addressed by the 16-bit address included with the instruction are transferred into the L register. The contents of memory at Addr+1 are transferred into the H register.

SHLD Addr(3)
Operation performed: $(L) \longrightarrow [Addr],$
$(H) \longrightarrow [Addr]$
The contents of the L register are transferred into the memory location specified by the 16-bit address included with the instruction. The contents of the L register are transferred into memory at Addr+1.

MOV D,M(1)
Operation Performed: $([HL]) \longrightarrow D$
The contents of the memory location whose address is specified by the HL register pair is transferred into the destination register.

MOV M,S(1)
Operation Performed: $(S) \longrightarrow [HL]$
The contents of the source register are transferred into the memory location whose address is specified by the HL register pair.

LDAX RD(1)
Operation Performed: $([RD]) \longrightarrow A$
The contents of the memory location whose address is specified by the register pair is transferred into the A register.

STAX RD(1)
Operation Performed: $(A) \longrightarrow [RD]$
The contents of the A register are transferred into memory at the address specified by the register pair.

POP RS(1)
Operation Performed: $([SP]) \longrightarrow RS_L,$
$([SP+1]) \longrightarrow RS_H,$
$(SP)+2 \longrightarrow SP$
The contents of the memory location addressed by the stack pointer are transferred into the low-order register of the register pair (ie., C, E, L or A). The stack pointer is incremented and the contents of the memory location now addressed are then transferred into the high-order register of the register pair (ie., B, D, H or Flags). The stack pointer is then incremented again.

PUSH RS(1)
Operation Performed: $(RS_H) \longrightarrow [SP\text{-}1],$
$(RS_L) \longrightarrow [SP\text{-}2]$
$(SP)\text{-}2 \longrightarrow SP$
The stack pointer is decremented by one. The contents of the high-order register of the register pair are then transferred into the memory location whose address is specified by the stack pointer. The stack pointer is then again decremented by one and the contents of the low-order register of the register pair are then transferred into the memory location now addressed by the stack pointer.

XTHL(1)
Operation Performed: $(HL) \longleftrightarrow ([SP])$
The contents of the HL register pair are exchanged with the contents of the top (last entered) elements in memory as addressed by the stack pointer.

Immediate data transfers

Immediate data transfers use data that is included with the instruction as the data source and an internal register, register pair or memory location as the data destination. By "included with the instruction", we mean data that is located in the memory location(s) immediately following the instruction. (Using this definition, the addresses included with some of the memory data transfers could be considered to be immediate addresses.)

MVI D,D8(2)
Operation Performed: $D8 \longrightarrow D$
The 8-bit immediate data is transferred into the destination register.

MVI M,D8(2)
Operation Performed: $D8 \longrightarrow [HL]$
The 8-bit immediate data is transferred into memory at the address specified by the HL register pair.

LXI RP,D16(3)
Operation Performed: $D16 \longrightarrow RP$
The 16-bit immediate data is transferred into the selected register pair.

I/O data transfers

I/O data transfers use input devices addressed by an I/O address as data sources and output devices addressed by I/O addresses as data destinations. As mentioned earlier, most individual I/O devices are either a data source or a data destination.

IN IO(2)
Operation Performed: $(IO) \longrightarrow A$
The data byte supplied by the input device specified by the I/O address is transferred into the A register.

OUT IO(2)
Operation Performed: $(A) \longrightarrow IO$
The contents of the A register are transferred to the output device specified by the I/O address.

Summary

The 8080 provides a large selection of data-transfer instructions for use in programming (97 unique machine codes). We will use these instructions to maneuver data and addresses around inside the computer.

In the next column in this series we will continue our discussion of instruction sets and the 8080 with a look at the important arithmetic/logic instructions. **R-E**