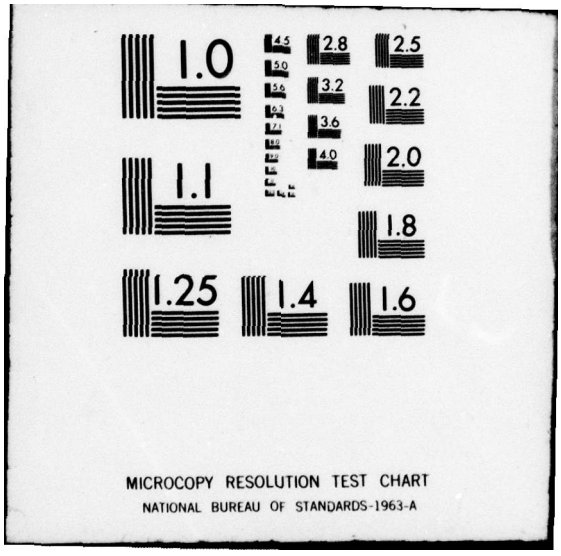
AD-A065 564 DEFENCE RESEARCH ESTABLISHMENT VALCARTIER (QUEBEC) F/G 9/2
AN AUTOMATIC PROGRAMMER FOR THE 2708/2704 ERASABLE PROGRAMMABLE--ETC(U)
FEB 79 B MONTMINY, J N BERUBE, R CARBONNEAU

UNCLASSIFIED DREV R-4131/79 NL

1 OF
AD
A065564

END
DATE
FILMED
5-79
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
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

NTIS REPRODUCTION
BY PERMISSION OF
INFORMATION CANADA

AD A065564

DDC FILE COPY

UNCLASSIFIED

CRDV RAPPORT 4131/79
DOSSIER: 3633A-010
FÉVRIER 1979

DREV REPORT 4131/79
FILE: 3633A-010
FEBRUARY 1979

LEVEL

DDC
RECEIVED
MAR 12 1979
C

AN AUTOMATIC PROGRAMMER FOR THE 2708/2704 ERASABLE PROGRAMMABLE READ-ONLY MEMORY

B. Montminy

J.N. Bérubé

R. Carbonneau

P. Côté

This document has been approved for public release and sale; its distribution is unlimited.

Centre de Recherches pour la Défense
Defence Research Establishment
Valcartier, Québec

BUREAU - RECHERCHE ET DEVELOPPEMENT
MINISTÈRE DE LA DÉFENSE NATIONALE
CANADA

RESEARCH AND DEVELOPMENT BRANCH
DEPARTMENT OF NATIONAL DEFENCE
CANADA

NON CLASSIFIÉ

79 03 09 018

CRDV R-4131/79
DOSSIER: 3633A-010

UNCLASSIFIED

(14) DREV R-4131/79
FILE: 3633A-010

(11) Feb 79

(12) 42p.

DDC
RECEIVED
MAR 12 1979
C

(6) AN AUTOMATIC PROGRAMMER FOR THE 2708/2704 ERASABLE PROGRAMMABLE READ-ONLY MEMORY

by

(10) B. Montminy, J.N. Bérubé, R. Carbonneau, P. Côté

CENTRE DE RECHERCHES POUR LA DEFENSE
DEFENCE RESEARCH ESTABLISHMENT
VALCARTIER
Tel: (418) 844-4271

Québec, Canada

February/février 1979

404 945

NON CLASSIFIE

79 03 09 018

RESUME

Ce document décrit un appareil conçu au CRDV et permettant la programmation de mémoires à lecture seulement (EPROM), effaçables à l'ultraviolet et reprogrammables électriquement, (Intel - 2708/2704). Un ordinateur à usage multiple reçoit les données à programmer et les transfère au programmeur par ligne téléphonique. Le programmeur utilise un système micro-informatique basé sur le microprocesseur Intel - 8080 dans lequel toutes les opérations sont commandées par un programme contenu dans une mémoire à lecture seulement. Cette technique permet une adaptation facile du programmeur à de nouveaux types de mémoire EPROM par un simple changement dans le programme du microprocesseur. (NC)

ABSTRACT

A programmer for the Intel 2708/2704, ultraviolet erasable and electrically reprogrammable read-only memory (EPROM) has been designed and built at DREV. Data to be programmed are fed into a general-purpose computer and then transferred to the programmer by means of a telephone line. The programmer is designed around a microcomputer system based on an Intel 8080 microprocessor in which all the operations are controlled by a program stored in a read-only memory. This type of design allows easy adaptation of the programmer for programming new EPROMs by simply changing the program of the microprocessor. (U)

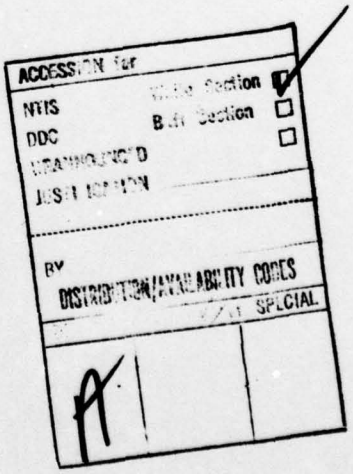


TABLE OF CONTENTS

1.0 INTRODUCTION

Recent developments in semiconductor technology have brought a new philosophy among digital circuit designers. Conventional digital systems (i.e. a particular arrangement of digital components performing a specific task) are now giving their place to more versatile and more powerful digital systems controlled by a program stored in a non-volatile semiconductor memory.

Three types of non-volatile semiconductor memories are commonly used: the read-only memory (ROM), the programmable read-only memory (PROM) and the erasable programmable read-only memory (EPROM). The ROM is factory programmed and very expensive in small quantities. The PROM is programmed by the user but has to be replaced by another device if data are erroneous. However, it is a good choice as a program-storage unit for small-quantity production. The EPROM has the advantage over the 2 other types of being easily modified in its content, which is often needed in prototype development.

The programmation of the new type of EPROM which could store up to 1K 8-bit words was time consuming and exposed to errors which required many hours to correct. For these reasons, an automatic EPROM programmer for the Intel 2708/2704 EPROM was designed and built at DREV by using a microprocessor. The programmer reduced a tedious one-day task to a five-minute easy and error-free task without the need for a costly microcomputer development system.

The work was performed at DREV in March and April, 1976 under PCN 33A10, "Improvement to equipment".

2.0 DESIGN PHILOSOPHY

As it was realized that an ever increasing number of DREV's projects were to rely on a microcomputer system, an assembler and a simulator for the Intel 8080 microprocessor were written first in APL (Ref.1) and are now also available in a speeded-up FORTRAN version (Ref. 2). The assembler gives a binary matrix which is the assembled program to be inserted into the EPROM for a new 8080 design. Whichever assembler is used (APL or FORTRAN), the matrix of the assembled program may easily be stored in an APL workspace. A programmer is thus required to transfer data from APL to the EPROM.

The design goal for the EPROM programmer was to ease the programming task by limiting to the minimum operator's interventions, and to achieve the highest programming speed possible. The programmer, to be described in section 3.0, is connected to DREV central computer via a telephone line by means of an acoustic coupler. Data to be programmed into the EPROM are assigned to the variable "A" of the addressed APL workspace. Log-on and workspace information can be obtained by communicating with the authors.

The programmer must take data in the APL workspace and program these into an Intel 2708/2704 EPROM. A considerable advantage of transferring data to be programmed from an APL workspace into the EPROM is the facility with which data can be modified. Of course, APL language is very flexible for manipulating vectors and matrices. When the data programmed into the EPROM has to be modified, the modifications are done in APL and the EPROM is reprogrammed with the new data after it has been erased.

Many reasons led to the selection of a microcomputer system to carry the programming task:

1) Since the data to be put in the EPROM may be a matrix of up to 1024 8-bit words, it appears that the only easy way for manipulating such data is a computer or a microcomputer. The latter was chosen.

2) The many programming steps needed for the Intel 2708-2704 EPROM are best handled by a microcomputer system. The programming operation description and the pertinent waveforms are given in Ref. 3.

3) A noteworthy advantage of using a microcomputer system in such a programmer is the possibility of programming other types of PROMs or EPROMs with data taken from an APL workspace by simply changing the program of the microprocessor and adding simple hardware to the system to satisfy the requirements of the new devices. This feature will be useful in the future when new types of EPROM emerge.

3.0 GENERAL DESCRIPTION

All the electronics for the programmer, power supplies included, is contained in a 12 x 7 x 4 in metal box (Figure 1a). Figure 1b shows the inside. Power supplies are on the left. The board containing the components forming the microcomputer system is wired using Wire-Wrap techniques. Components are mounted on Wire-Wrap sockets cemented on a perforated hardboard. This mounting technique ensures optimum packaging density.

3.1 Block diagram of the programmer

The microcomputer system of the programmer (Figure 2) is a standard system using the 8080 microprocessor as central processing unit (CPU). The key-parts of the system are:

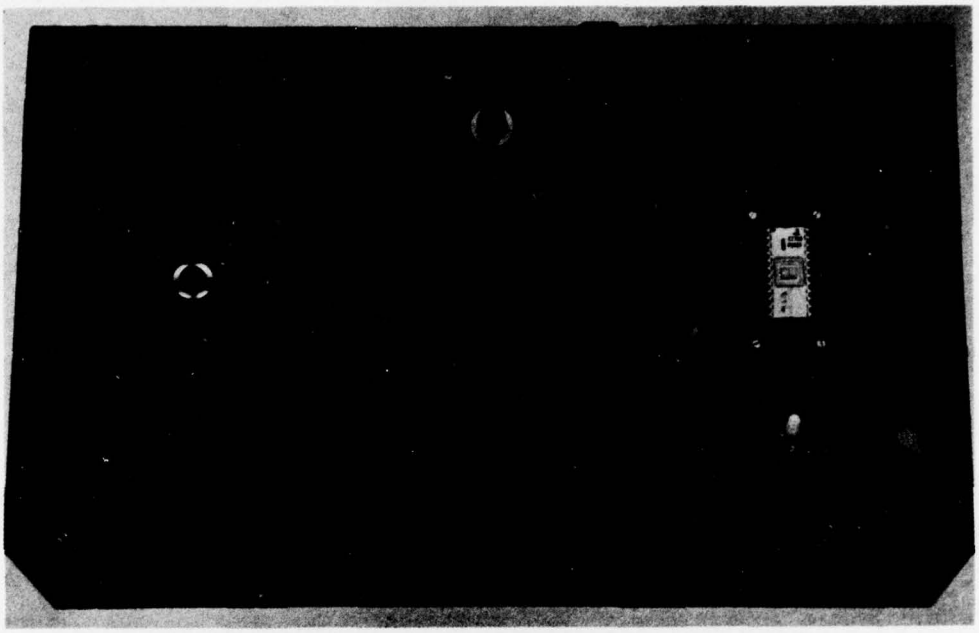


FIGURE 1a - EPROM programmer (outside view)

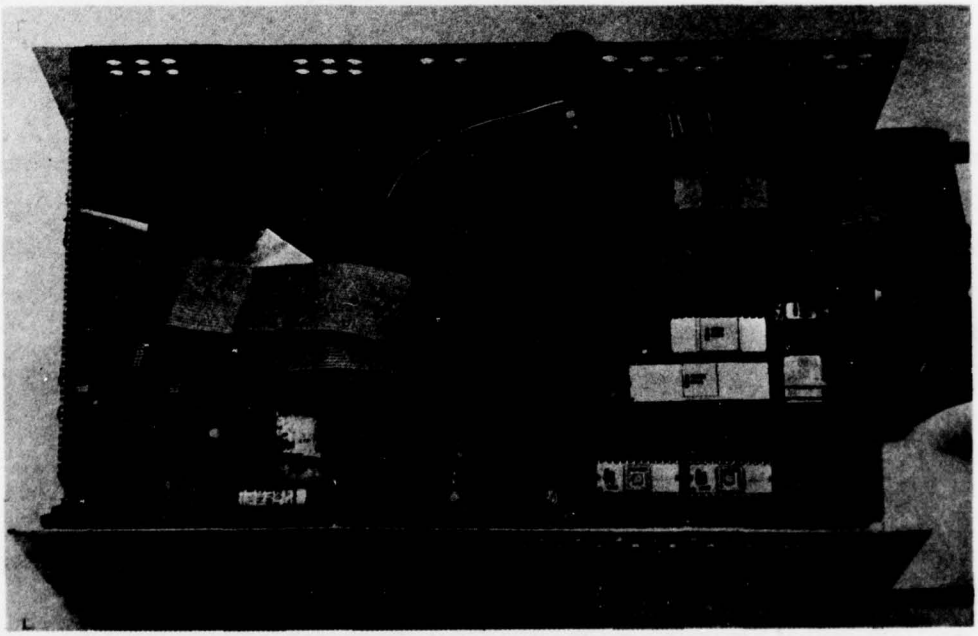


FIGURE 1b - EPROM programmer (inside view of the power supplies and electronic board

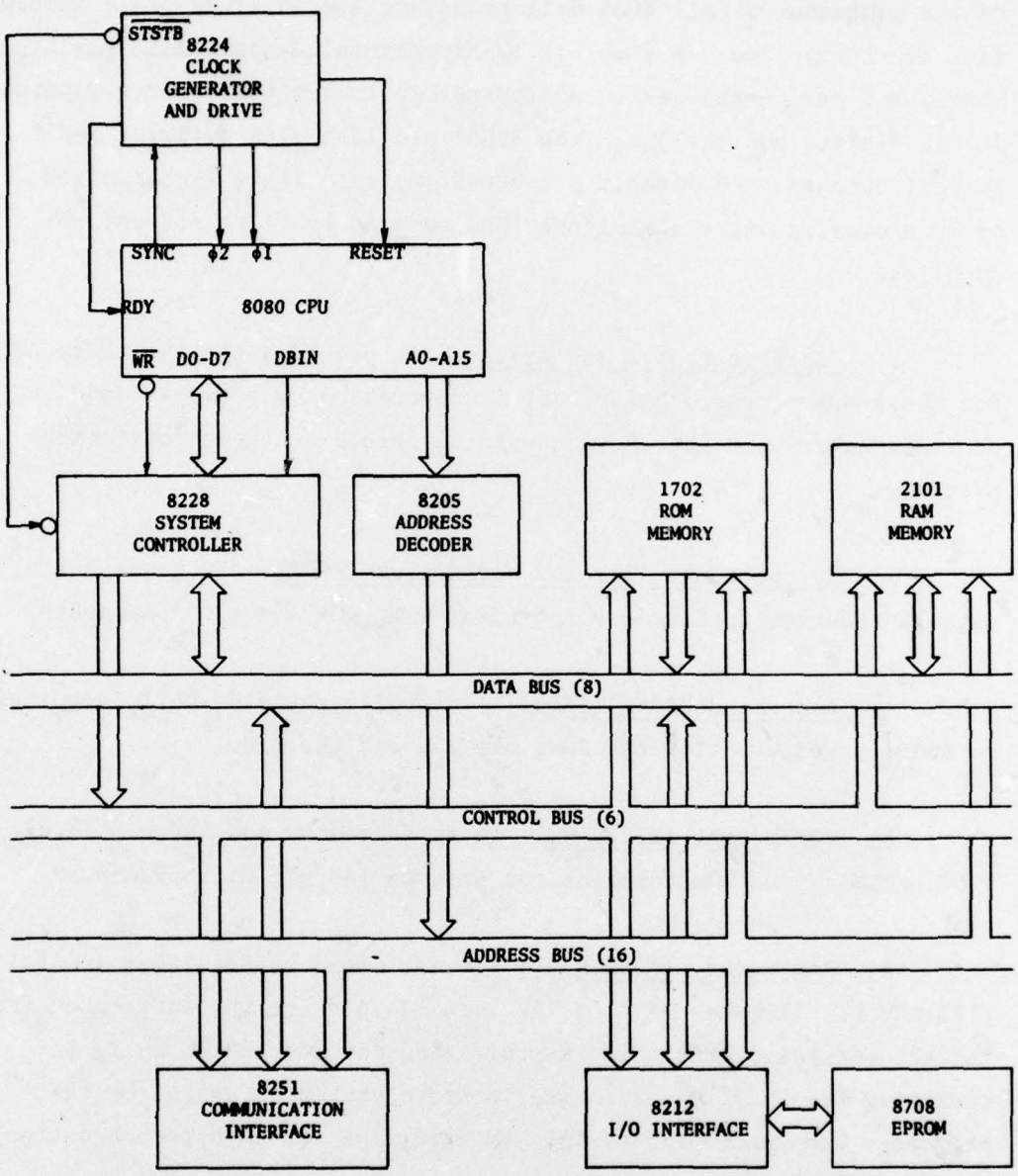


FIGURE 2 - Block diagram of the programmer

1) 8080 microprocessor: This complete 8-bit CPU is the heart of the programmer. All 8080 data transfers and internal-state information are controlled via an 8-bit, bidirectional 3-state data bus. Memory and peripheral-device addresses are transmitted over a separate 16-bit 3-state address bus. The 8080A provides also 6 timing and control outputs, and accepts 4 control inputs. It is synchronized by 2 nonoverlapping clock signals and powered by +12V, +5V and -5V supplies.

2) Clock generator and driver: It provides the two-phase clock for the 8080A microprocessor. It is controlled by a crystal and provides power-up reset, advance-status strobe and synchronization of ready.

3) System controller and bus driver (8228): It generates all signals required to directly interface RAM, ROM and 1/0 components.

4) Address decoder (8205): It performs the decoding required to address individually the ROM, the RAM and the 1/0.

5) ROM memory (512 x 8): It is formed by two 256 x 8 Intel 1702 EPROMs. The ROM contains the program for the microprocessor.

6) RAM memory (1280 x 8): It is formed by ten 256 x 4 Intel 2101 RAM's. 1K bytes of this RAM is used to store the data taken from the APL workspace before it is programmed into the EPROM while the remaining 256 locations are used to store variables needed by the program. The upper part of the RAM forms the stack where subroutine return addresses are stored.

7) Four 8-bit input/Output ports (8212): They consist each of 8-bit latches with 3-state output buffers along with control and device selection logic. The 1/0 ports transmit data from the data bus to the

system outputs. These outputs are the 2708/2704 EPROM and the light-emitting-diode (LED) indicators on the front panel of the programmer.

8) Programmable communication interface (8251): It is a Universal Synchronous/Asynchronous Receiver/Transmitter (USART) chip for serial data communications in microcomputer systems. It handles data communication between the general purpose DREV central computer and the 8080 microprocessor of the programmer.

3.2 Description of operations

The operations performed by the EPROM programmer are flow-charted in Figure 3. Upon depressing the reset push-button on the front panel, the execution of the 8080 program begins and the following steps are performed:

1) Initialization period

During this period, the stack pointer of the microprocessor is initiated, the LED indicators on the front panel are reset and the mode and command instructions are sent to the serial communication interface (8251).

2) Wait for a signal that indicates that the central computer is ready to receive the "SIGN ON"

This period lasts as much time as it takes to the operator to put the telephone handset on the acoustic coupler. Of course, the programmer detects that the computer is ready to accept the "SIGN ON" when it receives "LOGON PLEASE:".

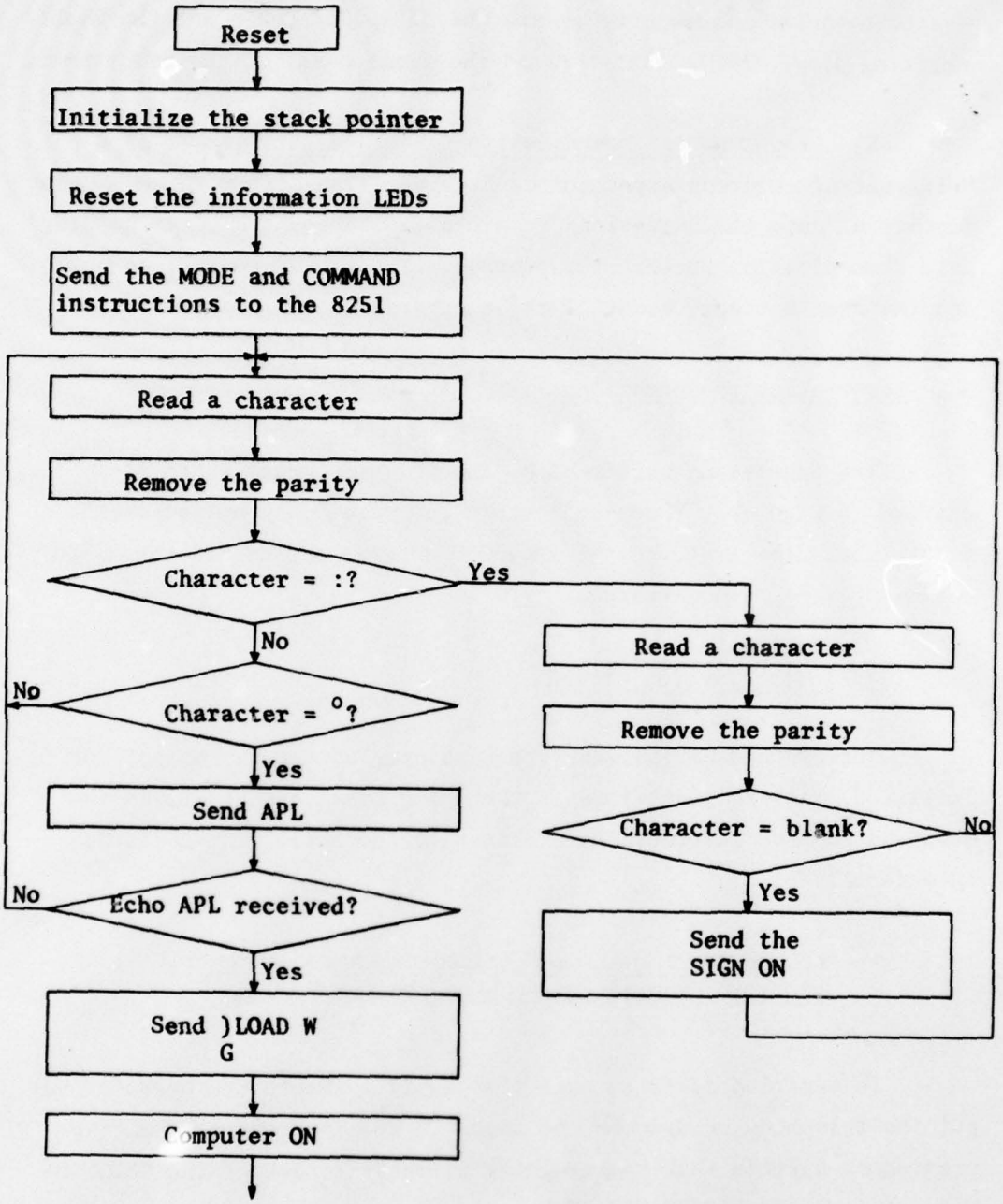


FIGURE 3 - Flow-chart of operations

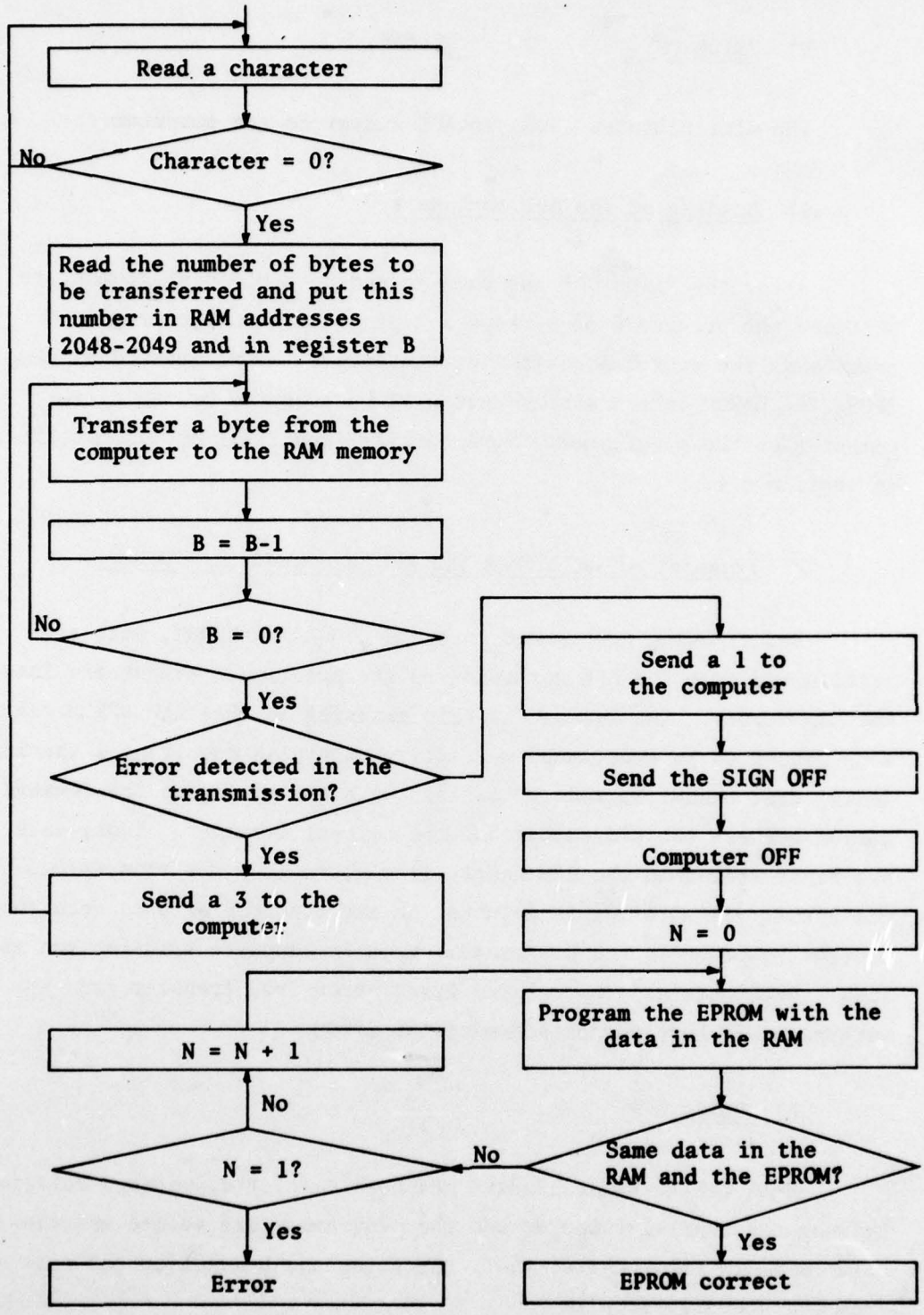


FIGURE 3 (continued)

3) "SIGN ON"

The microcomputer sends an APL number to the computer.

4) Loading of the APL workspace

After the "SIGN ON" has been accepted, a "LOAD W" command is sent and the program G of workspace W is executed. The program G transforms the vector A containing the data to be programmed by the 2708/2704 EPROM into a matrix that will be accepted by the microcomputer of the programmer. More details about this matrix are given in section 6.1.

5) Transfer of data from the APL workspace to the RAM

Before being programmed into the 2708/2704 EPROM, data are transferred from the APL workspace to the programmer and stored into the RAM memory. The reason for this transfer is that the EPROM takes many passes to be programmed correctly, each pass requiring a specific timing that cannot be made compatible with the timing of the communication between the programmer and the central computer. Thus, data are first stored in the RAM memory from where they are transferred to the EPROM. If an error is detected in the transfer of data from the central computer to the programmer, another complete transfer run is made. Data obtained through the first error-free transfer from APL workspace are kept for programming the EPROM.

6) "SIGN OFF"

When the transfer of data has been completed, no more relations between the central computer and the programmer are needed and the "SIGN OFF" is sent. After that, the telephone handset can be removed from the acoustic coupler.

7) Programmation of the 2708/2704 EPROM

Data are transferred from the RAM to the EPROM following the timing given in Ref. 3.

8) Comparison of the data in the EPROM with those in the RAM

This part is intended to check if the EPROM is well programmed. If RAM and EPROM contents are similar, the processor passes to the "end of operations" sequence, otherwise a second and last programmation run is attempted.

9) End of operations

At the end, if the content of the EPROM is the same as the content of the RAM, the LED indicator "END" is set on the front panel, indicating that all the operations of the programmer have been successful, and that they are completed. If contents are not the same, the LED indicator "ERROR" is set on the front panel showing that the data in the EPROM are not correct, which means that they are not similar to those transferred from the computer.

4.0 OPERATIVE INSTRUCTIONS

The operative instructions for programming the 2708/2704 EPROM are given in explaining what happens in the system when it is working. Indicators and switches functions on the front panel are also described.

4.1 Preparation of the data to be programmed into the 2708/2704 EPROM

The data to be programmed into the EPROM are put in the vector A of the APL workspace W of the APL number which the EPROM programmer refers to. This number can be obtained by contacting the authors.

If the data are to be programmed into a 2708 EPROM, A will be defined as: A ← a vector of 1 to 1024 elements (in decimal), each element being comprised between 0 and 255. Each element will be programmed into the EPROM according to its rank in the vector A, (i.e.: the first element of A will be programmed at the address 0 of the EPROM, the second at the address 1, etc.). If the vector A has less than 1024 elements, each element is programmed into the EPROM and the remaining positions of the EPROM are set to 255.

If the data are to be programmed into a 2704 EPROM which is a 512-byte EPROM, A will be defined by the following sequence:

```
A ← 1 to 512 elements (in decimal), each element being
                    comprised between 0 and 255,
A ← 512↑A, 512⍴255,
A ← A, A.
```

A is thus a 1024-element vector obtained by the catenation of 2 identical 512-element vectors. This is because the only difference between the 2708 and the 2704 EPROM from the programmation point of view is that the 2704 has one address line less than the other. In the 2708 EPROM, this address line selects between the first and the last 512 elements of the EPROM. If the data contained in the RAM are composed of 2 identical groups of 512 bytes, no error will be detected in the programmation of the 2704 by comparing the 512 bytes with the upper part of the RAM memory. This feature allows the programmation of a 2704 or a 2708 at will without changing or selecting anything on the programmer itself. Table I shows the definition of the data to be programmed into the 2708/2704 EPROM where A is the data to be programmed and W the name of the APL workspace.

TABLE I

Generation of vector A

APL NUMBER[1]

2708:)LOAD W
A ← 1 to 1024 elements (decimal) 0 ≤ each element ≤ 255
)SAVE

2704:)LOAD W
A ← 1 to 512 elements (decimal) 0 ≤ each element ≤ 255
A ← 512 ↑ A,512 ⍴255
A ← A, A
)SAVE

[1]: contact the authors

4.2 Description of indicators and switches

The "Reset" command initiates the operation of the programmer by resetting to zero the program counter of the microprocessor and by resetting the programmable communication interface (8251).

The "Program Test" switch allows the operator to disable the program pulse when it is in "test" position. If the program pulse is not fed to the 2708/2704 EPROM, no change can occur in its content. This feature allows checking the content of an EPROM by using only the verification part of the programmation of the EPROM. The normal position of the switch for programming an EPROM is "Program".

"Computer ON" and "Computer OFF" indicators tell the operator when the programmer is communicating with the computer. At the beginning of the programmer's operations, the 2 indicators are turned off. Then, the "Computer ON" indicator is turned on at SIGN ON while the "Computer OFF" indicator is turned on when the SIGN OFF is sent to the computer. At the same time, computer ON indicator is turned off. When the "Computer OFF" indicator is on, the telephone handset can be removed from the acoustic coupler.

The "Error" indicator tells that data programmed into the EPROM are different from those received from the computer. This may happen when an EPROM has not been completely erased.

The "END" indicator indicates a successful operation of the programmer. Data in the EPROM are then similar to those received from the computer.

The "Receive" and "Transmit" LEDs allow visual monitoring of the communication between the computer and the programmer.

The "Program pulse" LED is activated when program pulses go to the EPROM.

In addition to the above-mentioned switches and indicators, the power "ON/OFF" switch and the 24-pin socket for the 2704/2708 EPROM are mounted on the front panel.

5.0 HARDWARE

Detailed design of the system is given in Figures 4 to 8. This section will not explain the whole microcomputer system because this explanation can be found in many reference manuals. However, some peculiarities of this system are pointed out.

5.1 The microprocessor, the clock and the controller

Figure 4 shows the microprocessor, the clock and the controller which are standard parts of an 8080 microcomputer system. The cycle time for the 8080 is set to 0.5 μs by an 18-MHz crystal connected to the 8224 which divides this frequency by 9. However, the execution time of each instruction is increased by the introduction of "wait" states due to the high access-time of the memory modules used in the system. These wait states are introduced by means of a monostable multivibrator (14528-A) which forces the RDYIN of the 8224 to the "low" state at the beginning of each machine cycle.

The interrupt capability of the 8080 microprocessor is used in the system. The one-level interrupt is obtained by connecting the interrupting-device output (8251 RXRDY) directly to the INT input of the 8080. Since there is only one interrupt level, the beginning address of the interrupt routine is set to 56 by means of a 1-KΩ resistor connected between the INTA input of the 8228 and the + 12-V supply (Ref. 3). This arrangement introduces automatically a RST 7 instruction when an interrupt occurs.

5.2 The memory blocks

The memory blocks (Figure 5) are formed by two 1702 programmable read-only memories (EPROM), ten 2101 random access memories (RAM) and two one-of-eight decoders. The 1702 EPROMs were used because of the programmation facilities which were available at DREV. Each EPROM can store up to 256 bytes of program. The 2101 RAMs were used in spite of their separate 1/0 because 2111 RAMs were not available at the time of construction. These 1K RAMs, organized in 256 x 4 bits, are paired to form 256-byte blocks of memory.

The one-of-eight decoders (8205-1 and 2) are used to enable each device or block of memory following its predetermined address

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO DDC

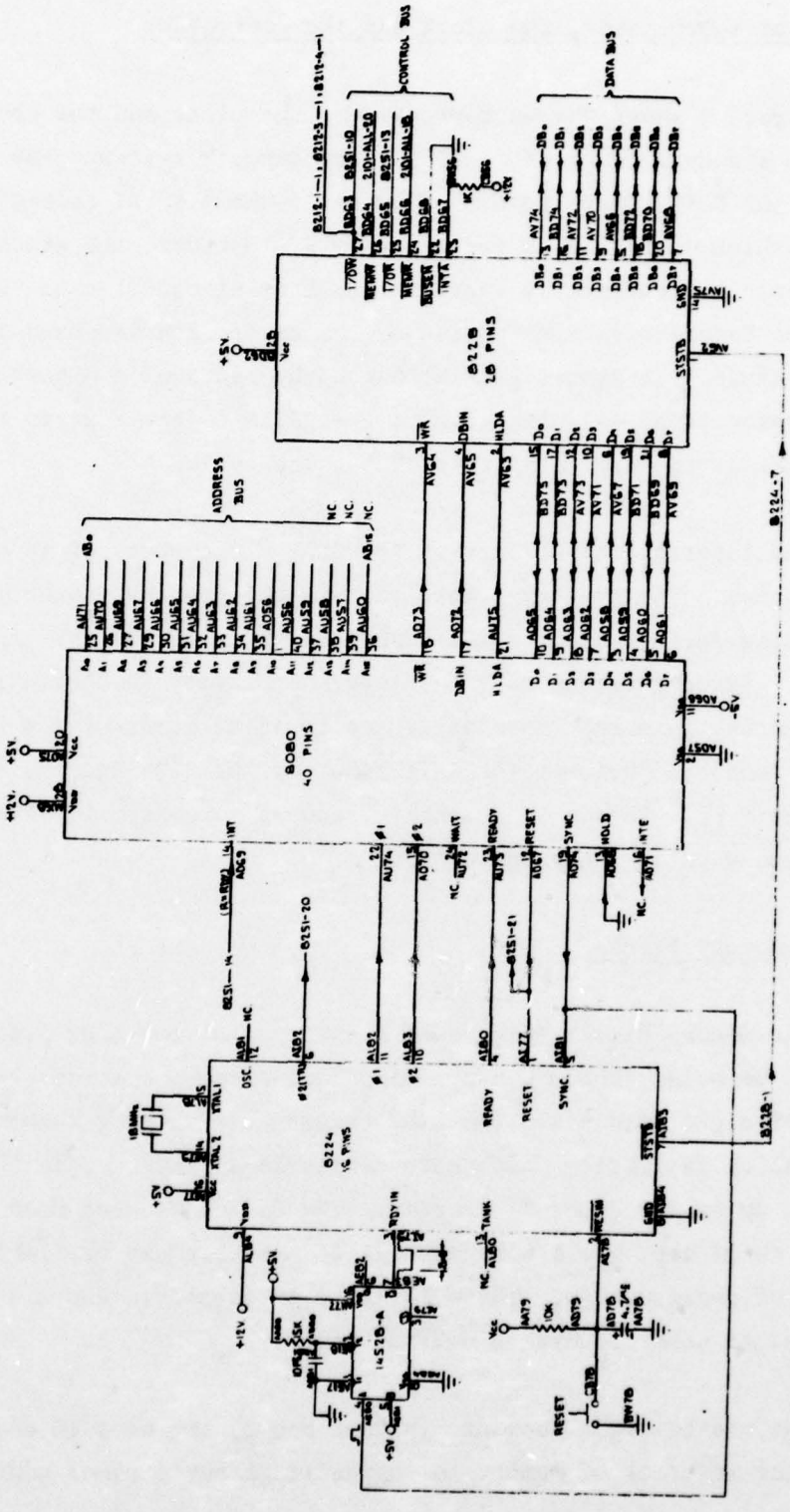


FIGURE 4 - Microprocessor, clock and controller

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO DDC

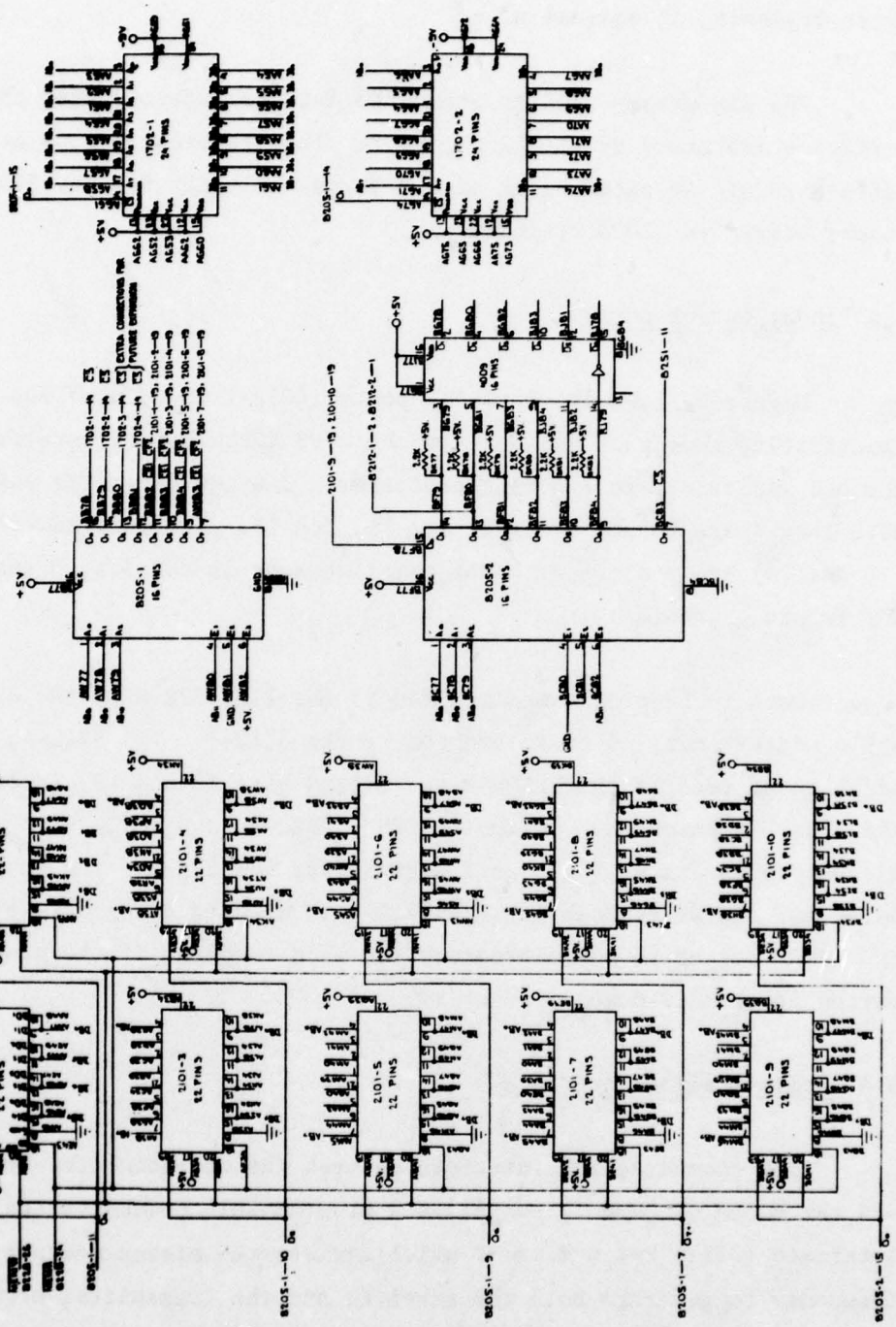


FIGURE 5 - The memory blocks

assignation. The program memory is stored in 2 blocks of 256 memory bytes beginning at address 0.

The RAM memory used to store the data transferred from the APL workspace is formed by the 2101-1 to 8. This 1K block begins at address 1024. An extra block of RAM forms the stack for the 8080 and covers addresses 2028 to 2303.

5.3 Input/output ports

Figure 6 shows the four 1/0 ports (8212-1 to 4) used for transferring addresses and data to the 2708 EPROM and/or interfacing the LED indicators to the microprocessor. 1/0 addresses for ports 8212-1 to 4 are respectively 9, 10, 11, and 12. Two more addresses (13 and 14) are present with the existing address decoding hardware for future expansion.

Data to be programmed are fed to the 2708 EPROM by the 8212-1 while address bits A0 to A7 are fed by the 8212-3. The 8212-4, in addition to feeding the EPROM with address bits A8 and A9, interfaces the LEDs "Computer ON", "Computer OFF", "ERROR" and "END" with the microprocessor as well as the 2 signals for the EPROM: the program pulse and the write enable. The 8212-2 is used to bring data from the EPROM output to the microprocessor when checking of the programmation process is done.

5.4 Communication interface

The communication interface between the microcomputer system and the MODEM (Figure 7) comprises a programmable communication interface (8251) and a divider which divides the microprocessor clock frequency to generate both the receiver and the transmitter clocks for the 8251. The divider is formed by counters 74193-1, 74193-2, 7490 and the one-shot 14528-B. This arrangement of counters performs a

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO DDC

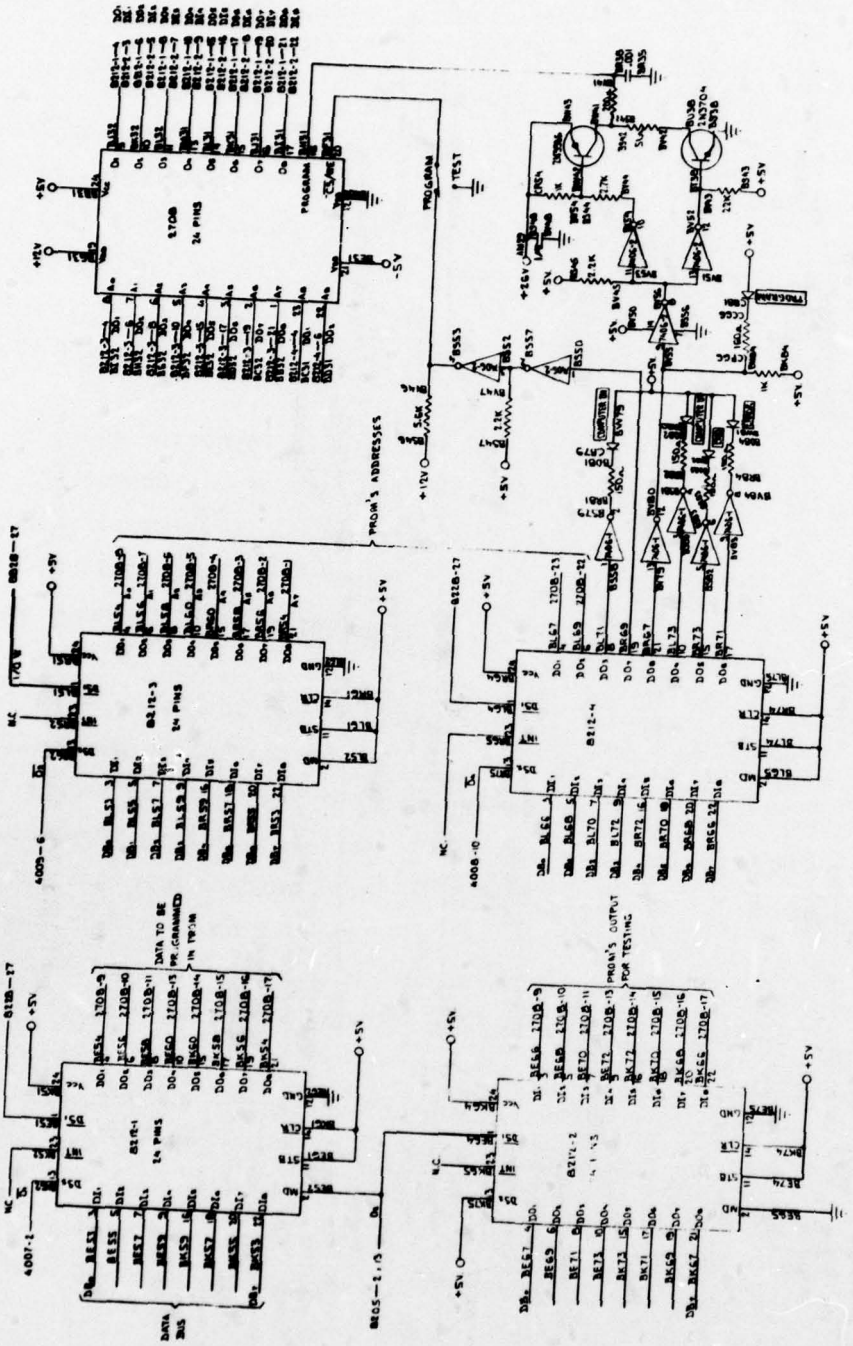


FIGURE 6 - Input/Output interface and 2708 EPROM

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO DDC

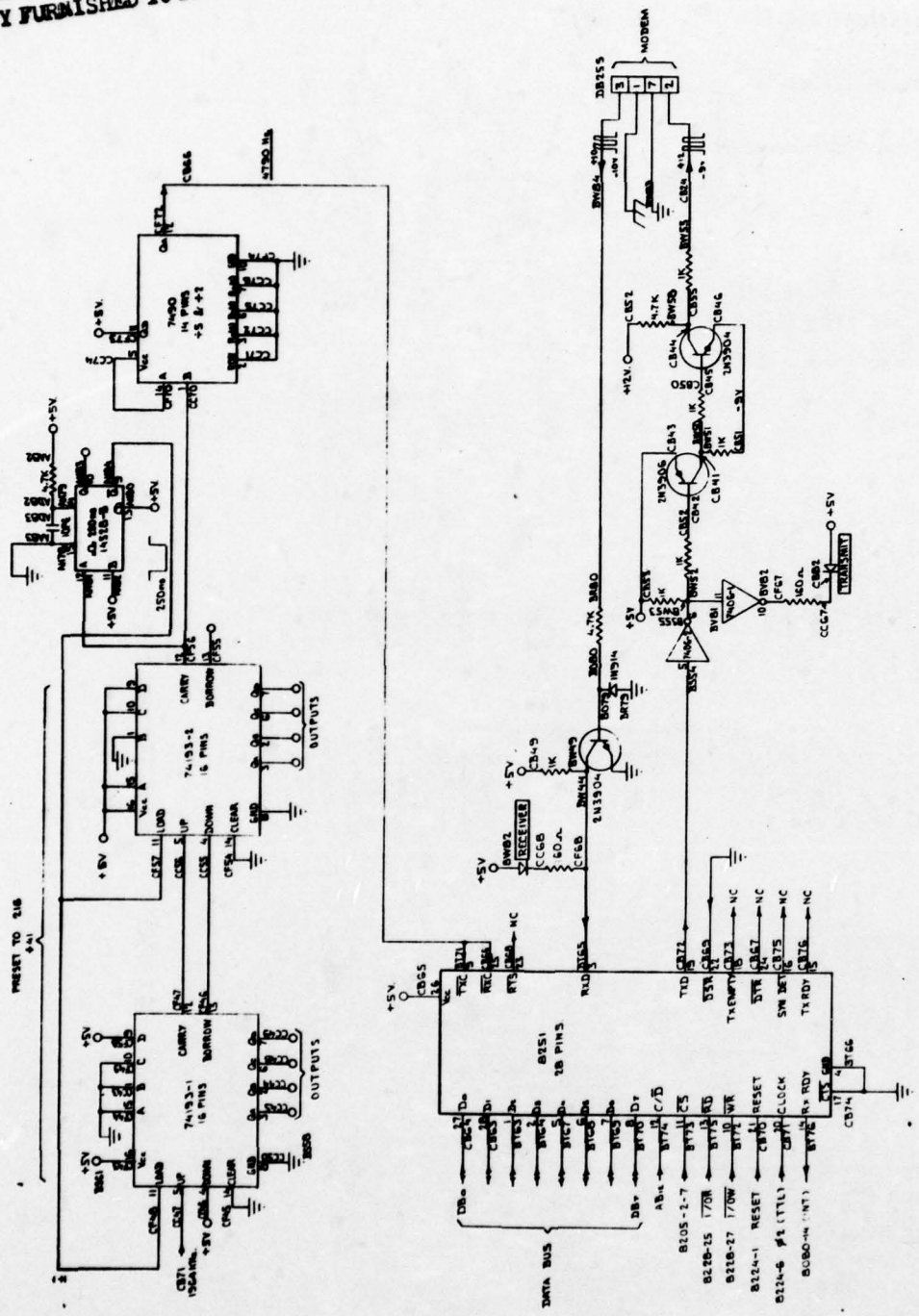


FIGURE 7 - The communication interface

division by 420, such that the 1964 KHz microprocessor frequency provides the 4790-Hz clock feeding the transmitter and the receiver clocks of the 8251. This frequency is close to 4800 Hz, which is 16 times the baud rate at which the communication is made with the computer. Output voltage-levels to the MODEM are provided by the communication interface which also accepts input voltage-levels of the MODEM.

5.5 Power supplies

Five voltage sources (Figure 8) are needed to supply the circuits of the programmer. The 8080 microprocessor itself and the 2708 EPROM need 3 sources: +5V, +12V and -5V while the 1702 PROMs need +5V and -9V. All the other circuits need only a single 5-V source. However, the program pulse for programming the 2708 EPROM must be 26V.

6.0 SOFTWARE

The necessary software to program the EPROM with data contained in a vector A of an APL workspace is divided in 2 parts. The first part is an APL program which first rearranges the vector A into a matrix and then transfers it to the programmer. The second part of the software is a machine language program for the microcomputer system of the programmer. This program is stored into PROMs.

6.1 The APL part of the software

Due to some difficulties of synchronization encountered with the 8251 communication interface, data in vector A could not be transferred directly to the programmer. Each element of vector A had to be transmitted in 2 halves. At the same time, a parity bit was added to each transmitted half to enable checking transmission by the

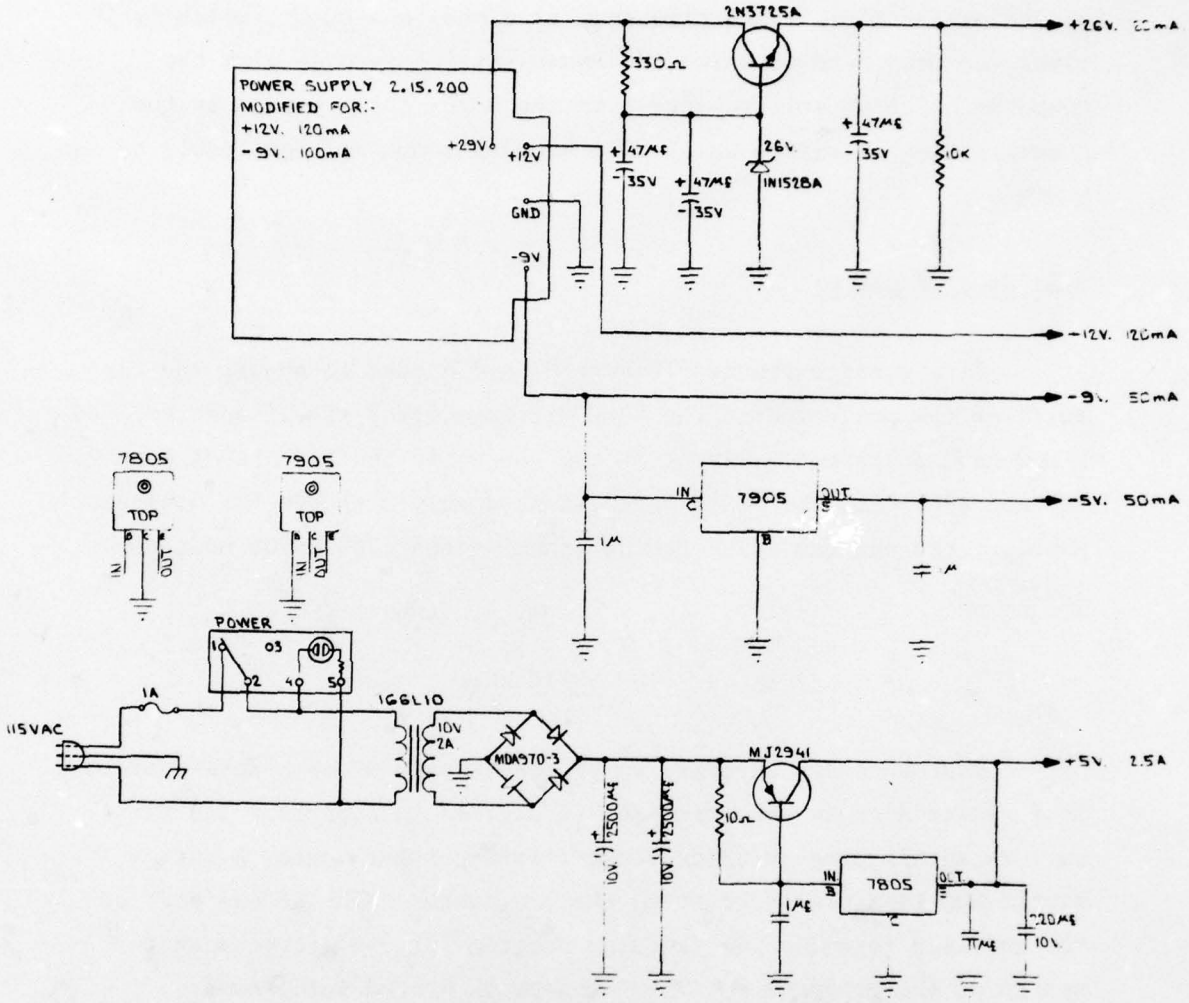


FIGURE 8 - The power supplies

microcomputer system. The format of each transmitted half is: $001PB_3B_2B_1B_0$ where B_0 to B_3 is the byte and P is an odd parity bit.

The APL program G which performs the rearrangement of the data in A is given in Table II. The output of this program as transmitted to the programmer is given in Table III. The beginning of transmission is indicated by a nul (0) while the end is indicated by the character $\alpha(65)$.

6.2 The microprocessor program

The microprocessor program flow-charted in Figure 3 is listed in the Appendix. Its 184 assembler instructions give 339 bytes of machine language program that are stored in two 1702 EPROMs. Comments in the listing of the Appendix give a description of the program with some points to be now emphasized.

At line 3 of the program, the mode instruction is sent to the 8251 communication interface. The mode instruction, in this case, sets the number of stop bits to 1, the character length to 8 and the baud rate factor to 16X. At line 7, the command instruction resets all error flags and enables receive mode. The first HLT instruction is encountered at line 9. The program stops there until the user puts the telephone handset on the acoustic coupler. Then, the program waits for the automatic "LOGON" request which is detected by the presence of a character ":" followed by a space, it sends the "SIGN ON" and waits for the character "o" afterwhat it responds by sending the characters "A", "P", "L" and carriage return. From this moment on, the computer has to send the echo "APL" within the next 16 characters in order to complete the link. If this fails, the programmer waits for another "LOGON" request from the computer and attempts another linking.

TABLE II

APL program G

```
     ∇G;X;S;B;C;D
[1]  J1:B←TERM 4013
[2]   S←0⎕'MOD'
[3]   X←⍬,256 256⍴⍳D←A,(1024-⍴A)⍴255
[4]   C←,⍉16 16⍴X,D
[5]   C←⍉(0≠2 2 2 2⊤512↑C),(0≠2 2 2 2⊤512↑512↓C),(0≠2 2
      2 2⊤512↑1024↓C),0≠2 2 2 2⊤1536↓C
[6]   ⎕←0,(2⊥⍉0,0,1,(=/C),C),65
[7]   B←TERM 3010
[8]   →(1≠⎕)/J1
      ∇
```

TABLE III

Output of program G transmitted to the programmer

Nul (0)	Indicates the beginning of transmission
MSB_{4-7}	
MSB_{0-3}	Number of words to transfer (16 bits)
LSB_{4-7}	
LSB_{0-3}	
$D1_{4-7}$	Data 1
$D1_{0-3}$	
$D2_{4-7}$	Data 2
$D2_{0-3}$	

DL_{4-7}	Last data
DL_{0-3}	
α(65)	Indicates the end of transmission

Each transmitted data except the first and the last words is of the form 0 0 1 P B_3 B_2 B_1 B_0 where B_0-B_3 are 4 transmitted bits while P is the odd parity.

Once communication is established between the central computer and the programmer, a "LOAD W" command is sent to the computer and the APL program G is executed while the programmer is waiting for a character "NUL" (line 83). After this first character, the number of words to be sent by the computer are read (lines 87 to 92) and put in memory addresses 2048-2049. Data are then transferred from the computer to the programmer (lines 94 to 100) and a 1 is sent to the computer to indicate that no error has been detected. Should an error be detected, a 3 has to be sent to the computer (line 48) and another transfer attempt is required. After the "SIGN OFF" has been sent (line 111), data are transferred from the RAM memory to the EPROM (lines 115 to 142) according to the timing given in the Appendix and comparison is made between the RAM memory and the EPROM. If contents are not similar, another programming process is tried. After 2 unsuccessful runs, the "Error" indicator is turned on and the program is stopped.

7.0 CONCLUSION

An original idea developped at DREV allows use of APL facilities for arranging data to be programmed into a 2708/2704 EPROM. Generation of data or their modification is easily done in APL before it is programmed automatically into the EPROM. If few EPROMs are available, modification of data and reprogrammation of another EPROM are done well within 10 minutes whereas it would have taken more than one day for a skilled technician to do the same job with a manual programmer.

The automatic programmer has proved to be very useful and even absolutely necessary in many DREV projects since its realization. Most DREV projects involve development of new systems which require many modifications before giving satisfactory performances. Some of these systems use a microprocessor which executes a program stored in an EPROM. For these projects, the automatic programmer is of prime

importance since a FORTRAN program is available to translate man-readable statements into machine-understandable code and to simulate any program written for the Intel 8080 microprocessor. The FORTRAN program outputs a vector A ready to be programmed into a 2708/2704 EPROM.

To give some examples of DREV projects that take advantage of the programmer, let us mention the Panoramic Passive Optical Tracker project in which four 2708 EPROMs are used, 3 to store a program for an 8080 microprocessor and the fourth to contain data for a character generator. A Gap Measuring Device uses two 2708 EPROMs to store 16-bit instructions that are executed by a CMOS logic circuit similar to a microprocessor. A 2708 EPROM is also used in a Radar Data Acquisition System. Finally, a new design for a Passive Infrared Intrusion Alarm based on an 8080 microprocessor uses a 2708 EPROM for storing the machine language program.

In the future, new EPROMs, more powerful than the 2708/2704, will become available and modifications will have to be performed onto the programmer to accomodate the new coming devices. However, the concept of transferring data from a general-purpose computer to a microcomputer system will remain the same and can be applied to other systems. For instance, a system similar to the microprocessor system of the programmer could interface one or many devices or instruments to a general-purpose computer. Data from the device could be transferred to the computer by means of a phone line for immediate processing. This principle could also be used to transfer medical data from a small locality to an important medical center equipped with a computer as well as scientific data taken during a field trial. The number of possibilities is limited only by the imagination.

8.0 ACKNOWLEDGMENTS

We would like to thank especially A. Blanchard of Data Systems Division who gave us all necessary information about the communication with the DREV central computer and made all the arrangements for getting the programmer accepted as an input for that computer. Thanks are also due to M. Lessard who built the programmer and participated actively in the debugging.

9.0 REFERENCES

1. Bérubé, J.N., "An Assembler and a simulator for the 8080 microprocessor", DREV M-2402/76, June 1976, UNCLASSIFIED

2. Bérubé, J.N., Carbonneau, R., Montminy, B., Côté, P., "An improved version of an assembler and simulator for the 8080 microprocessor", DREV R-4124/78, UNCLASSIFIED

3. Intel Corp., "8080 Microcomputer Systems User's Manual", July 1975.

APPENDIX

MICROPROCESSOR PROGRAM

```
1    LXI SP, 2303        Initiate the stack pointer (SP) to
                         the last RAM address
2    MVI A, 78           Send the mode instruction to the
                         8251 communication interface
3    OUT 31
4    XRA A
                         Reset LED indicators
5    OUT 12
6    MVI A, 20
                         Command instruction for the
                         communication interface
7    OUT 31
8    ATT: EI             Enable interrupt
9    HLT                 Wait for a character from the
                         computer
10   ANI 127             Remove the parity bit
11   CPI 62              Compare with ":"
12   JZ ESPA             Yes:  send the "SIGN ON"
13   CPI 74              No:  Compare with "o"
14   JNZ ATT             No:  wait for another character
15   LXI, H 466
16   MVI E, 4            Yes:  Send:  APL + carriage return
17   CALL WRITE
18   MVI E, 240          E = -16 APL echo must be received
19   ESS:  INR E         Within 16 caracters
20   JZ ATT
```

```
21      HLT
22      ANI 127
23      CPI 97              Compare to "A"
24      JNZ ESS
25      HLT
26      JMP INTER
27      RUPT:  IN 15
28      EI                  Interrupt routine:  input one
                            character
29      RET
30      ESPA:  HLT          Wait for a character from the
                            computer
31      ANI 127             Remove the parity
32      CPI 32              Compare with a space
33      JNZ ATT             No
34      LXI H, 492
35      MVI E, 15           Yes:  send the "SIGN ON"
36      CALL WRITE
37      JMP ATT             Read other characters
38      RESET:  POP B
39      INX SP
40      INX SP
41      B2:  HLT            Read a character
42      CPI 65              Compare to "α" (last character of
                            the transmission)
43      JNZ BZ
44      HLT                 Wait for 2 more characters
```

```
45      HLT
46      MVI E, 2            Send a 3 to indicate that an error
47      LXI H, 464          has occurred in the transmission
48      CALL WRITE
49      JMP ATT1
50      READ:  PUSH B
51      HLT                 Wait for a character
52      ADI 0               Check the parity.  If an error
53      JPO RESET           occurs, jump to reset
54      RAL
55      ORA A
56      RAL
57      ORA A               Put the first half of the work to
                            be
58      RAL                 read in register B
59      ORA A
60      RAL
61      MOV B, A
62      HLT                 Wait for the second half of the
                            word.
63      ADI 0               Check the parity
64      JPO RESET
65      ANI 15              Assemble the 2 halves of the word
66      ORA B               and put them in register A
67      POP B
68      RET
```

```
69      INTER:  ANI 127
70      CPI 112                 Compare with "p"
71      JNZ ESS
72      HLT
73      ANI 127
74      CPI 108                 Compare with "L"
75      JNZ ESS
76      MVI E, 18               If APL echo has been received
                                correctly,
77      LXI H, 471              a "LOAD W" command is sent and the
                                program.
78      CALL WRITE              G is executed
79      MVI E, 235
80      MVI A, 4                Turn ON the "computer ON" LED
81      OUT 12
82      ATTI:  LXI H, 2048
83      HLT
84      ANA A                   Wait for character "O"
85      JNZ ATTI
86      CALL READ
87      APR:  MOV C, A          Read the number of words to be
                                sent by
88      MOV M, A                the computer and put this number in
89      INX H                   register BC and in RAM locations
                                2048
90      CALL READ               and 2049
91      MOV B, A
```

```
92      MOV M, A
93      LXI H, 1024             First RAM address
94      CONT:  CALL READ
95      MOV M, A
96      INX H                   Transfer of data from the computer
97      DCX B                   to the RAM memory
98      MOV A, B
99      ORA C
100     JNZ CONT
101     HLT                     Wait for 2 characters
102     HLT
103     MVI E, 2
104     LXI H, 489              Send a 1 with a carriage return
                                to say
105     CALL WRITE              to the computer that data has been
                                transferred with success
106     HLT                     Wait for another character
107     OFF:  DI                Disable interrupt
108     MVI A, 8                Turn the computer "ON LED OFF"
                                and turn
109     OUT 12                  the computer "OFF LED ON"
110     LXI, H, 507
111     MVI E, 5                Send the "SIGN OFF"
112     CALL WRITE
113     LXI H, 2050             Put RAM location 2050 to 0
114     MVI M, 0
```

```
115     TRANS:  MVI E, 255          Number of passes to program the
                                    EPROM
116     TRA:  LXI B, 1024           First RAM address
117     LHLD 2048                   Number of words to program.
118     MVI A, 72                   CS/WE = 1 and write pulse = 0
119     OUT 12
120     PROGRAM:  MOV A, C          Addresses A0-A7 to the EPROM
121     OUT 11
122     MOV A, B
123     XRI 68                      Addresses A8-A9 to the EPROM.
124     OUT 12
125     LDAX B                      Word to program.
126     OUT 9
127     MOV A, B
128     XRI 196                     Send the write pulse to the EPROM
129     OUT 12
130     MVI A, 30
131     WAIT:  DCR A                Write pulse time ≈0.6 ms
132     JNZ WAIT
133     MOV A, B
134     XRI 68                      Remove the write pulse
135     OUT 12
136     INX B
137     DCX H                       Check if all the words have been
138     MOV A, L                    programmed into the EPROM
139     ORA H
```

```
140     JNZ PROGRAM
141     DCR E                   Check if all the passes have been
142     JNZ TRA                 executed
143     LHLD 2048               Number of words to
144     MOV D, H                program in DE
145     MOV E, L
146     LXI H, 1024
147     COM1:  MOV A, L
148     OUT 11
149     MOV A, H                Send address to the EPROM
150     XRI 4
151     OUT 12
152     IN 10                   Compare the content of the EPROM
153     CMP M                   with the content of the RAM
154     JNZ ERREUR              An error is detected if contents
                                are not similar
155     DCX D
156     MOV A, E                If the content of the EPROM is
157     ORA D                   similar to the content of the RAM,
158     JZ FIN                  the operation is a success
159     INX H
160     JMP COM1
161     ERREUR:  LXI H, 2050    If there is an error in the
                                programmation
162     INR M                   of the EPROM, RAM location is
                                increased
163     MOV A, M
```

```
164     CP1 2                   If an error is detected for the
                                first
165     JNZ TRANS               time, the programmation steps are
                                repeated
166     MV1 A, 32               After 2 errors, the "ERROR" LED is
167     OUT 12                  turned ON and the program is
                                stopped
168     HLT
169     FIN:  MVI A, 16
170     OUT 12                  The "END" LED is turned ON to
                                indicate
171     HLT                     a success in the operations
172     WRITE:  MVI A, 1        Command instruction to the
                                communication
173     OUT 31                  interface (write mode)
174     W1:  MOV A, M           Send a character to the 8251
175     OUT 15
176     WR:  IN 31              Check the status for transmitter
                                ready
177     RAR
178     JNC WR
179     INX H
180     DCR E                   Check if all words have been sent
181     JNZ W1
182     MVI A, 21               Return the 8251 to the read mode
183     OUT 31
184     RET
```

CRDV R-4131/78 (NON CLASSIFIE)

Bureau - Recherche et Développement, MDN, Canada.
CRDV, C.P. 880, Courcelette, Qué. G0A 1R0

"Programmeur automatique pour les mémoires de type EPROM 2704/2708"
par B. Montminy, J.N. Bérubé, R. Carbonneau, P. Côté

Ce document décrit un appareil conçu au CRDV et permettant la programmation de mémoires à lecture seulement (EPROM), effaçables à l'ultraviolet et reprogrammables électriquement, (Intel - 2708/2704). Un ordinatuer à usage multiple reçoit les données à programmer et les transfère au programmeur par ligne téléphonique. Le programmeur utilise un système micro-informatique basé sur le microprocesseur Intel - 8080 dans lequel toutes les opérations sont commandées par un programme contenu dans une mémoire à lecture seulement. Cette technique permet une adaptation facile du programmeur à de nouveaux types de mémoire EPROM par un simple changement dans le programme du microprocesseur. (NC)

CRDV R-4131/78 (NON CLASSIFIE)

Bureau - Recherche et Développement, MDN, Canada.
CRDV, C.P. 880, Courcelette, Qué. G0A 1R0

"Programmeur automatique pour les mémoires de type EPROM 2704/2708"
par B. Montminy, J.N. Bérubé, R. Carbonneau, P. Côté

Ce document décrit un appareil conçu au CRDV et permettant la programmation de mémoires à lecture seulement (EPROM), effaçables à l'ultraviolet et reprogrammables électriquement, (Intel - 2708/2704). Un ordinatuer à usage multiple reçoit les données à programmer et les transfère au programmeur par ligne téléphonique. Le programmeur utilise un système micro-informatique basé sur le microprocesseur Intel - 8080 dans lequel toutes les opérations sont commandées par un programme contenu dans une mémoire à lecture seulement. Cette technique permet une adaptation facile du programmeur à de nouveaux types de mémoire EPROM par un simple changement dans le programme du microprocesseur. (NC)

CRDV R-4131/78 (NON CLASSIFIE)

Bureau - Recherche et Développement, MDN, Canada.
CRDV, C.P. 880, Courcelette, Qué. G0A 1R0

"Programmeur automatique pour les mémoires de type EPROM 2704/2708"
par B. Montminy, J.N. Bérubé, R. Carbonneau, P. Côté

Ce document décrit un appareil conçu au CRDV et permettant la programmation de mémoires à lecture seulement (EPROM), effaçables à l'ultraviolet et reprogrammables électriquement, (Intel - 2708/2704). Un ordinatuer à usage multiple reçoit les données à programmer et les transfère au programmeur par ligne téléphonique. Le programmeur utilise un système micro-informatique basé sur le microprocesseur Intel - 8080 dans lequel toutes les opérations sont commandées par un programme contenu dans une mémoire à lecture seulement. Cette technique permet une adaptation facile du programmeur à de nouveaux types de mémoire EPROM par un simple changement dans le programme du microprocesseur. (NC)

CRDV R-4131/78 (NON CLASSIFIE)

Bureau - Recherche et Développement, MDN, Canada.
CRDV, C.P. 880, Courcelette, Qué. G0A 1R0

"Programmeur automatique pour les mémoires de type EPROM 2704/2708"
par B. Montminy, J.N. Bérubé, R. Carbonneau, P. Côté

Ce document décrit un appareil conçu au CRDV et permettant la programmation de mémoires à lecture seulement (EPROM), effaçables à l'ultraviolet et reprogrammables électriquement, (Intel - 2708/2704). Un ordinatuer à usage multiple reçoit les données à programmer et les transfère au programmeur par ligne téléphonique. Le programmeur utilise un système micro-informatique basé sur le microprocesseur Intel - 8080 dans lequel toutes les opérations sont commandées par un programme contenu dans une mémoire à lecture seulement. Cette technique permet une adaptation facile du programmeur à de nouveaux types de mémoire EPROM par un simple changement dans le programme du microprocesseur. (NC)

DREV R-4131/78 (UNCLASSIFIED)

Research and Development Branch, DND, Canada.
DREV, P.O. Box 880, Courcelette, Que. G0A 1R0

"An Automatic Programmer for the 2708/2704 Erasable Programmable Read-Only Memory" by B. Montminy, J.N. Bérubé, R. Carbonneau, P. Côté

A programmer for the Intel 2708/2704, ultraviolet erasable and electrically reprogrammable read-only memory (EPROM) has been designed and built at DREV. Data to be programmed are fed into a general-purpose computer and then transferred to the programmer by means of a telephone line. The programmer is designed around a microcomputer system based on an Intel 8080 microprocessor in which all the operations are controlled by a program stored in a read-only memory. This type of design allows easy adaptation of the programmer for programming new EPROMs by simply changing the program of the microprocessor. (U)

DREV R-4131/78 (UNCLASSIFIED)

Research and Development Branch, DND, Canada.
DREV, P.O. Box 880, Courcelette, Que. G0A 1R0

"An Automatic Programmer for the 2708/2704 Erasable Programmable Read-Only Memory" by B. Montminy, J.N. Bérubé, R. Carbonneau, P. Côté

A programmer for the Intel 2708/2704, ultraviolet erasable and electrically reprogrammable read-only memory (EPROM) has been designed and built at DREV. Data to be programmed are fed into a general-purpose computer and then transferred to the programmer by means of a telephone line. The programmer is designed around a microcomputer system based on an Intel 8080 microprocessor in which all the operations are controlled by a program stored in a read-only memory. This type of design allows easy adaptation of the programmer for programming new EPROMs by simply changing the program of the microprocessor. (U)

DREV R-4131/78 (UNCLASSIFIED)

Research and Development Branch, DND, Canada.
DREV, P.O. Box 880, Courcelette, Que. G0A 1R0

"An Automatic Programmer for the 2708/2704 Erasable Programmable Read-Only Memory" by B. Montminy, J.N. Bérubé, R. Carbonneau, P. Côté

A programmer for the Intel 2708/2704, ultraviolet erasable and electrically reprogrammable read-only memory (EPROM) has been designed and built at DREV. Data to be programmed are fed into a general-purpose computer and then transferred to the programmer by means of a telephone line. The programmer is designed around a microcomputer system based on an Intel 8080 microprocessor in which all the operations are controlled by a program stored in a read-only memory. This type of design allows easy adaptation of the programmer for programming new EPROMs by simply changing the program of the microprocessor. (U)

DREV R-4131/78 (UNCLASSIFIED)

Research and Development Branch, DND, Canada.
DREV, P.O. Box 880, Courcelette, Que. G0A 1R0

"An Automatic Programmer for the 2708/2704 Erasable Programmable Read-Only Memory" by B. Montminy, J.N. Bérubé, R. Carbonneau, P. Côté

A programmer for the Intel 2708/2704, ultraviolet erasable and electrically reprogrammable read-only memory (EPROM) has been designed and built at DREV. Data to be programmed are fed into a general-purpose computer and then transferred to the programmer by means of a telephone line. The programmer is designed around a microcomputer system based on an Intel 8080 microprocessor in which all the operations are controlled by a program stored in a read-only memory. This type of design allows easy adaptation of the programmer for programming new EPROMs by simply changing the program of the microprocessor. (U)