SOLAR

6940B Multiprogrammer
Users Guide
for the
Hewlett•Packard Interface Bus

with calculators

9820A
9821A
9830A

HP Computer Museum
www.hpmuseum.net

For research and education purposes only.

ERRATA SHEET
6940B/9820A, 21A, 30A Users Guide
for the
Hewlett-Packard Interface Bus
Users Guide Part No. 59500-90003

Make all corrections in the Users Guide according to the following errata.

On page 5-6, in Example 7, change line no. 13Ø in the 9830 sample program to read: IF A4#64 THEN 15Ø

On pages 5-10 and 5-11, DATA OUTPUT - 9830 to 6940 - VERIFICATION program, make the following changes:

1. Delete line no. 372.
2. Change line nos. 45Ø and 478 to read:
   FOR I = 1 TO 98

On page 5-13, GATE/FLAG VERIFICATION program (9830 calculator) change line no. 7Ø4 to read:
CMD "?U7" , "ØØØ2ØTAT"

On page 5-14, 59500A LISTEN MODE VERIFICATION program (9820A/21A calculator), add the following NOTE to explain the ⌗ symbol in line nos. 1Ø and 17.

NOTE

The ⌗ symbol is obtained by pressing the blank key on the left keyboard corresponding to 42_8 (See Appendix D in 9820A/21A HP-IB Users Guide, 59300-90001).

On page 5-16, DATA OUTPUT VERIFICATION program (9820A/21A calculator), insert the following line numbers (1Ø through 53) after line number 9. These lines are missing from the original program. Renumber the remaining lines in the program starting with line no. 54 (original line no 1Ø becomes line no. 54, etc.). Also, change original line no. 78 on page 5-18 to read: CMD "?U7" , "@" ⊢

```
10:
0→C→A→B⊢
11:
"X";2↑B*10↑A→X⊢
12:
FMT "@",FXD *.0;
WRT 13,X⊢
13:
PRT "ONLY 6940 B
IT"⊢
14:
PRT C,"ON?"⊢
15:
SPC 2⊢
16:
STP ⊢
17:
B+1→B;IF 3=B;
GTO "Y"⊢
18:
C+1→C⊢
19:
GTO "X"⊢
20:
"Y";A+1→A;0→B⊢
21:
IF 4=A;GTO "P"⊢
22:
C+1→C⊢
23:
GTO "X"⊢
24:
"P";CMD "?U7","A
"⊢
25:
PRT "ONLY 6940 B
IT"⊢
26:
PRT "12 ON?"⊢
27:
SPC 2⊢
28:
STP ⊢
29:
CMD "?U7","B"⊢
30:
PRT "ONLY 6940 B
IT"⊢
31:
PRT "13 ON?"⊢
32:
SPC 2⊢
33:
STP ⊢
34:
CMD "?U7","D"⊢
35:
PRT "ONLY 6940 B
IT"⊢
36:
PRT "14 ON?"⊢
37:
SPC 2⊢
38:
STP ⊢
39:
CMD "?U7","H"⊢
40:
PRT "ONLY 6940 B
IT"⊢
41:
PRT "15 ON?"⊢
42:
SPC 2⊢
43:
STP ⊢
44:
CMD "?U7","G7777
"⊢
45:
PRT "ONLY 6940 B
IT"⊢
46:
PRT "15 OFF?"⊢
47:
SPC 2⊢
48:
STP ⊢
49:
CMD "?U7","K7777
"⊢
50:
PRT "ONLY 6940 B
IT"⊢
51:
PRT "14 OFF?"⊢
52:
SPC ⊢
53:
STP ⊢
```

Add to DATA OUTPUT VERIFICATION Program on page 5-16.

On page 5-21, GATE/FLAG VERIFICATION program (9820A/21A calculator), change line no. 9 to read:
CMD "?U7" , "0002ØT@T" ⊢

On page 6-14, 69321B Sample Test Program, change line no. 8Ø of 9830 program to read: OUTPUT (13,7Ø) "B"A2"T";

On page 6-18, in Example 15, delete the first line in both sample programs.

On page 6-18, in Example 16, add the following line at the beginning of both sample programs: CMD"?U7" , "0004ØT"

On page 6-29, 69335A Sample Test program, change fourth line in 9820/21A program to read: IF A > –.ØØ1; JMP 2

On page 6-31, 69370A Sample Test Program, change third line in 9820/21A program to read: A/.ØØ5 → A

On page 6-43, in Example 42, change both sample programs to read: CMD "?U7" , "0004ØTB7777T"

On page 6-51, 69436A Sample Test Program, change the sixteenth line in the 9820/21A program to read:
IF C = 1; GTO "END"

On page 6-54, after step 1 of paragraph 6-250 (top left), change program line for 9820/21A to read:
IF (A > –.ØØ1) (A ⩽ 4Ø); JMP 4

On page 6-54, after step 2 of paragraph 6-250 (top left), change program line for 9820/21A to read: A/.Ø25 → A

On page 6-55, in Example 57, change ninth line of 9820/21A sample program to read: C/ (1* .Ø2) → A

On page 6-61, Example 62, changes are required in the programs to prevent the possibility of power supplies shorting together when using a 69330A Relay Output Card to scan their output voltages. To avoid this possibility, it is first necessary to open the relay contacts before programming the next sequence of contact closures. This is accomplished by making the following changes to the sample programs in Example 62.

9830A Calculator Program

1. Change line no. 4Ø to read: CMD "?U7" , "ØØ16ØTAT"
2. Add line 145 as follows: GOSUB 2ØØØ

9820/21A Calculator Program

1. Change the fourth program line to read:
   CMD "?U7" , "ØØ16ØTAT"
2. Add the following line before the END statement of the program: GSP "SPOLL"

On page 6-62, add the following to explain the changes in the sample programs of Example 62.

Explanation:

.
.
.

4Ø Establishes the output mode with DTE, SYE, and TME on. TME is included in the control word followed by AT which transmits zeros to the relay output card. TME ensures that the relay contacts have sufficient time to open before programming the next sequence of contact closures.

.
.
.

145 Clear the service request which was set by the control word containing TME in line no. 4Ø.

On page 6-63, in Example 63, change eighth line of 9820/21A sample program to read:
FMT Z, "T0016ØT0024ØTAX" , FXT * .Ø; WRT 13

On page A-1, paragraph A-10, change fourth line of example (4) to read: $0 \times 2^4 = 0 \times 16 = 0$

The following "octal-to-decimal" and "decimal-to-octal" subroutines can be used in place of those listed on pages A-6 and A-7 of Appendix A. The new routines execute faster than the original routines.

9830A Subroutines:

```
2ØØØ REM OCTAL TO DECIMAL CONVERSION
2Ø1Ø A2=A–2* INT(A/1Ø)–16* INT (A/1ØØ)–128*INT
(A/1ØØØ)
2Ø2Ø RETURN

24ØØ REM DECIMAL TO OCTAL CONVERSION
241Ø IF A > –Ø.5 THEN 243Ø
242Ø A=4Ø96 + A–1E–Ø8
243Ø A=INT(A+Ø.5)
244Ø A2=A+2*INT(A/8)+2Ø*INT(A/64)+2ØØ*INT(A/512)
245Ø RETURN
```

9820/21A Subroutines:

```
4: "DEC";A–2*INT(A/1Ø)–16*INT(A/1ØØ)–128*INT
(A/1ØØØ) → R2 ⊢
5: RET ⊢

11: "OCT"; IF A > –.5; JMP 2 ⊢
12: 4Ø96+A–1E–8 → A ⊢
13: INT (A+.5) → A ⊢
14: A+2*INT(A/8)+2Ø*INT(A/64)+2ØØ*INT (A/512) → R2 ⊢
15: RET ⊢
```

12-9-75

6940B MULTIPROGRAMMER USERS GUIDE

for the

HEWLETT-PACKARD INTERFACE BUS

HP Part No. 59500-90003

July 1975

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

Chapter I
INTRODUCTION

1-1 SCOPE

1-2 This guide provides programming, installation, and verification procedures for the 6940B multiprogrammer system as used on the Hewlett-Packard Interface Bus (HP-IB) under control of an HP Programmable Calculator (9830A, or 9820/21A). Also included in this guide are condensed circuit descriptions of the 6940/41B multiprogrammer (Chapter II) and the 59500A Multiprogrammer Interface (Chapter III). Although these descriptions do not provide a complete hardware analysis showing every signal and circuit, they are quite comprehensive and provide the user with a good understanding of the multiprogrammer system capabilities. The intent of this guide, therefore, is to provide all of the information necessary to program and use the 6940B multiprogrammer and minimize the need for constant referrals to other documents (manuals, data sheets, etc.,).

1-3 The following is an abbreviated listing of all of the major information contained in this guide together with a brief description concerning the scope of this material.

(1) Installation. Chapter IV contains all of the information necessary for interconnecting the units, setting the address switches and turning on the equipment in the proper sequence.

(2) Programming. Chapter V, Paragraphs 5-4 through 5-31 contains the fundamentals of programming the multiprogrammer system; while Chapter VI contains the detailed programming sequences associated with each one of the plug-in cards. In addition, Chapter II, Paragraphs 2-40 through 2-65, include a theoretical explanation of the 6940B programming sequences, complete with flow diagrams. Note that the information in this guide relates only to programming requirements that are unique to the multiprogrammer system. It is assumed that the user is familiar with the fundamentals of programming a 9830A, 9820A or 9821A calculator.

(3) Verification. Chapter V, Paragraphs 5-32 through 5-51, contain verification procedures for the 59500A/6940B mainframe (excluding any plug-in cards or 6941B Extender mainframes). These procedures verify that the major functions of the 59500A/6940B can be programmed. The tests are done on a GO/NO-GO basis, with detailed isolation procedures left to the Operating and Service Manuals for the 59500A and 6940B.

Sample test programs for each plug-in card are also included throughout Chapter VI in the individual card section. Again, detailed troubleshooting procedures must be obtained from the Operating and Service Manual for the particular card.

(4) Condensed Circuit Descriptions, 6940B and 59500A. As mentioned previously, simplified descriptions of these two units are given in Chapters II and III. Of course, users that are familiar with the multiprogrammer system can bypass Chapters II and III and go directly to the programming information in Chapters V and VI. However, users that are not familiar with the multiprogrammer or have complex programming requirements, are strongly urged to read these two chapters.

(5) Utility Subroutines. Another of the programming aids in this guide are the utility subroutines of Appendix A. These subroutines can be called from the users main program to perform frequently required operations.

(6) Number System Theory. Appendix A also contains a description of the number systems applicable to the multiprogrammer system. Included are explanations of the decimal, octal, binary, and BCD systems and how to convert from one system to another.

1-4 RELATED PUBLICATIONS

1-5 Other multiprogrammer literature that will be helpful to the user, includes the Operating and Service Manuals for the multiprogrammer mainframes (6940B and 6941B), the interface unit (59500A) and for each type of plug-in card. Each manual contains troubleshooting instructions, circuit theory, circuit diagrams and a replaceable parts list. Information on a more general level is contained in the 6940B multiprogrammer brochure (5952-3956 D) and the 59500A-HP-IB data sheet (5952-3977).

1-6 Calculator publications include the users guides (59300-90002 for 9830A or 59300-90001 for 9820/21A) and the Operating and Programming Manuals (09830-90001 or 09820-90001).

Chapter II
MULTIPROGRAMMER 6940B/6941B DESCRIPTION

2-1 INTRODUCTION

2-2 In the most general sense, the 6940B/6941B multiprogrammer units function as a multi-channel bi-directional interface between a controller and the "real world" environment to which command signals are sent, and from which status signals are received. For the purposes of this discussion, the controller may be a calculator (properly interfaced with the 6940B) or a computer which is capable of providing a sixteen bit word directly to the 6940B. When the multiprogrammer is controlled by a calculator on the HP-IB, a 59500A interface unit must be used to convert the eight bit ASCII coded output of the calculator into the sixteen bit format required by the multiprogrammer. Operation of the 59500A multiprogrammer interface unit is described in Chapter III.

2-3 Communication between the multiprogrammer and the external environment is realized via plug-in input/output (I/O) cards, of which more than thirty different types now exist. Data transfers between the multiprogrammer mainframe and the I/O cards are digital (twelve data bits); transfers between the cards and external devices can be either digital (again, twelve data bits) or analog (voltages, currents, etc.). Cards with analog I/O capability have on-card converters that permit them to transmit data to and from the mainframe in digital form.

2-4 Slot receptacles for fifteen cards are provided in both the 6940B and 6941B mainframes. As many as sixteen mainframes (a 6940B master and up to fifteen 6941B slaves) may be chained together so as to provide a maximum possible total of 240 I/O channels, any one of which may be selectively accessed by the controller.

2-5 Data Transfers

2-6 Data transfers between the 6940B and the controller take the form of a sixteen bit output word (from the controller to the multiprogrammer) and a thirteen bit (twelve data bits plus one status bit) return data word (from the multiprogrammer to the controller). From the standpoint of the I/O card-mainframe interface all data transfers are twelve bit; the controller, however, is required to output a sixteen bit word in order to bring about the twelve bit communication. This reduction in word length provides the mechanism by which the multiprogrammer functions as a multiplexer, effectively expanding a single duplex I/O channel into many bi-directional I/O channels. The cards themselves generally act in one capacity (input or output) to the exclusion of the other, but either card type may be used in any I/O channel.

2-7 Data Transfer Synchronization

2-8 Two additional signal lines ($\overline{GTE}$ and $\overline{FLA}$) facilitate a "handshake", permitting synchronization of data transfers. The gate signal ($\overline{GTE}$) is sent by the controller to indicate that it has valid data for the multiprogrammer to accept, while the flag signal ($\overline{FLA}$) permits the multiprogrammer to indicate that it (1) is processing data; and (2) has completed processing. Figure 2-1 shows the sequence of a handshake cycle. A data transfer cycle is not finished unless the handshake is also completed, in which case the controller may begin a new transaction. In some operating modes, completion of the handshake may take considerable time, particularly when compared to the period required by the controller to execute a program step. If this situation is known to exist, the controller may be directed to proceed with other portions of the program (data reduction, communication with other peripherals, etc.), returning periodically to test the progress of the handshake. Once a transfer cycle is begun by means of the gate signal, failure of the system to provide both edges of flag will always create a "hang-up" situation in which the controller will be unable to initiate new communications with the multiprogrammer unless the handshake is terminated by special programming commands.

2-9 Handshake capability may be extended through the multiprogrammer mainframe to the I/O card in certain modes of operation thereby altering the period (but not the sequence) of the handshake to suit the timing requirements of the particular card. In addition some cards permit the handshake to be extended to the external device in order to ensure that data transfers are accurate and complete. In either case, the mainframe no longer automatically returns the flag signal, but instead relinquishes control to the card and/or the external device. Figure 2-2 shows an example of this technique.

2-10 Extender Units

2-11 Any number (none at all or up to fifteen total) of 6941B extender units may be included in a system as required. It is not possible however, to use a 6941B without

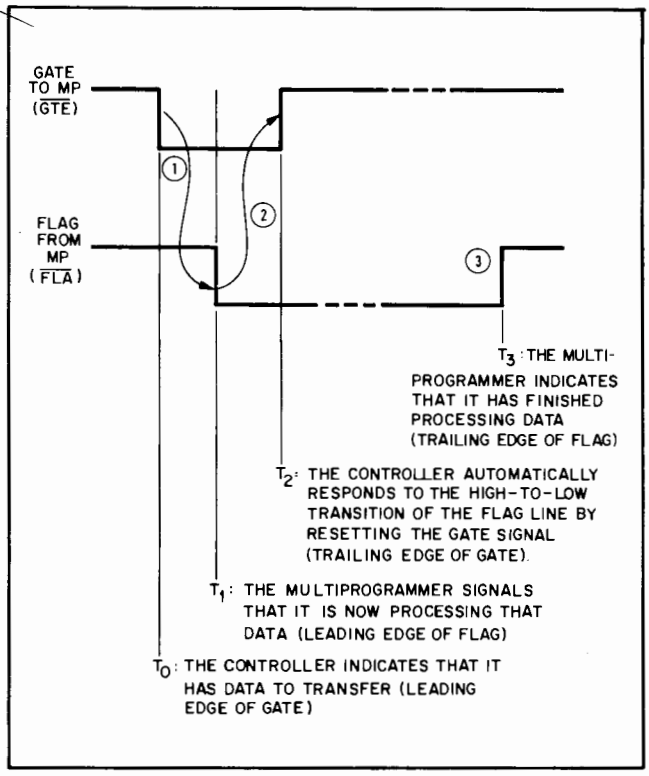


Figure 2-1. Handshake Sequence

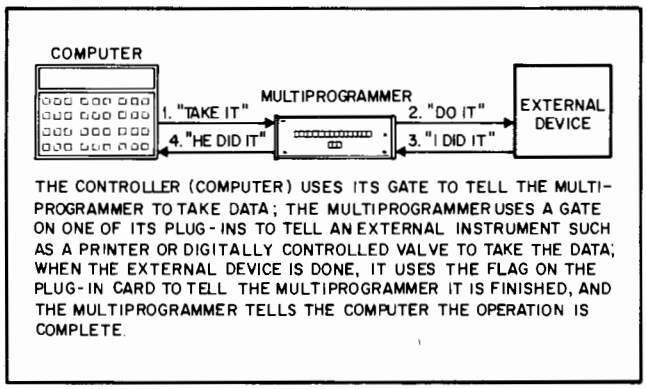


Figure 2-2. Extended Handshake

a 6940B in the system. In addition, it is necessary that the 6940B be the first unit in the chain. A cabling diagram showing how a 6940B and one or more 6941B's are chained together in a system is provided in Chapter IV.

2-12 Slot and Unit Addresses

2-13 Whether or not 6941B extender units are used, each card slot automatically assumes a two part address which uniquely specifies its location within the system. A card slot within either type mainframe has a fixed address (slot ØØ through slot 14) and may be accessed by means of a four bit binary code ($\overline{B12}$ through $\overline{B15}$) as shown in Table 2-1. The sixteenth address (Control Word) accesses circuitry which is present in the 6940B mainframe only, and permits the controller to specify a unit select address, thereby providing the second part of the total address needed to access a particular I/O channel.

2-14 Like the slot address, the unit select address is specified by a four bit binary code, ($\overline{BØØ}$ through $\overline{BØ3}$), but in this case all sixteen possible combinations are interpreted as unit addresses. The address of a particular unit is automatically established by its position in the chain; the first 6941B becomes unit Ø1, the second becomes unit Ø2, etc. The 6940B mainframe is always unit ØØ and is specified when $\overline{BØØ}$ through $\overline{BØ3}$ are logical "0's".

2-15 Once a data transfer cycle with a control word is completed, both the mode of operation and the unit select address have been stored by the 6940B mainframe, and subsequent operations will be directed towards the particular unit addressed. Slot addresses, designated in succeeding words output by the controller, cannot access card slots except in the previously selected unit unless a new control word is sent specifying a different unit address. Similarly, if it is desired that the operating mode be changed but not the unit address, the new control word must re-specify the particular unit address.

2-16 BLOCK DIAGRAM DESCRIPTION

2-17 At this point it is convenient to introduce both the functional block diagram as shown in Figure 2-3, and the concept of word format. Figure 2-3 shows the 6940B mainframe logic, the controller, a typical I/O card, and the significant connections between these segments of the total system. Although several signal lines between the mainframe and the I/O card have been omitted in order to simplify the block diagram, it remains faithful to the functional mechanisms of data transfer and it may be assumed that the card could be located in any I/O channel in the system. The signal lines shown between the multiprogrammer and the controller correspond exactly to the hardware with the single exception that the SYE (system enable) jumper connection is not included. This feature serves only to provide an interlock on the SYE control line such that a cable must be connected between the multiprogrammer and the controller before SYE can be sent to the cards. Certain cards are de-activated to a "safe" state in the absense of the SYE signal.

2-18 Figure 2-4 shows the format, or structure, of a data word. The data word is the most general of three formats used for the sixteen bit output word sent from the controller to the multiprogrammer. $\overline{B12}$ through $\overline{B15}$ specify the card slot address, while the remaining twelve bits are used to transfer data. Note that the hardware organization as shown in the block diagram complements

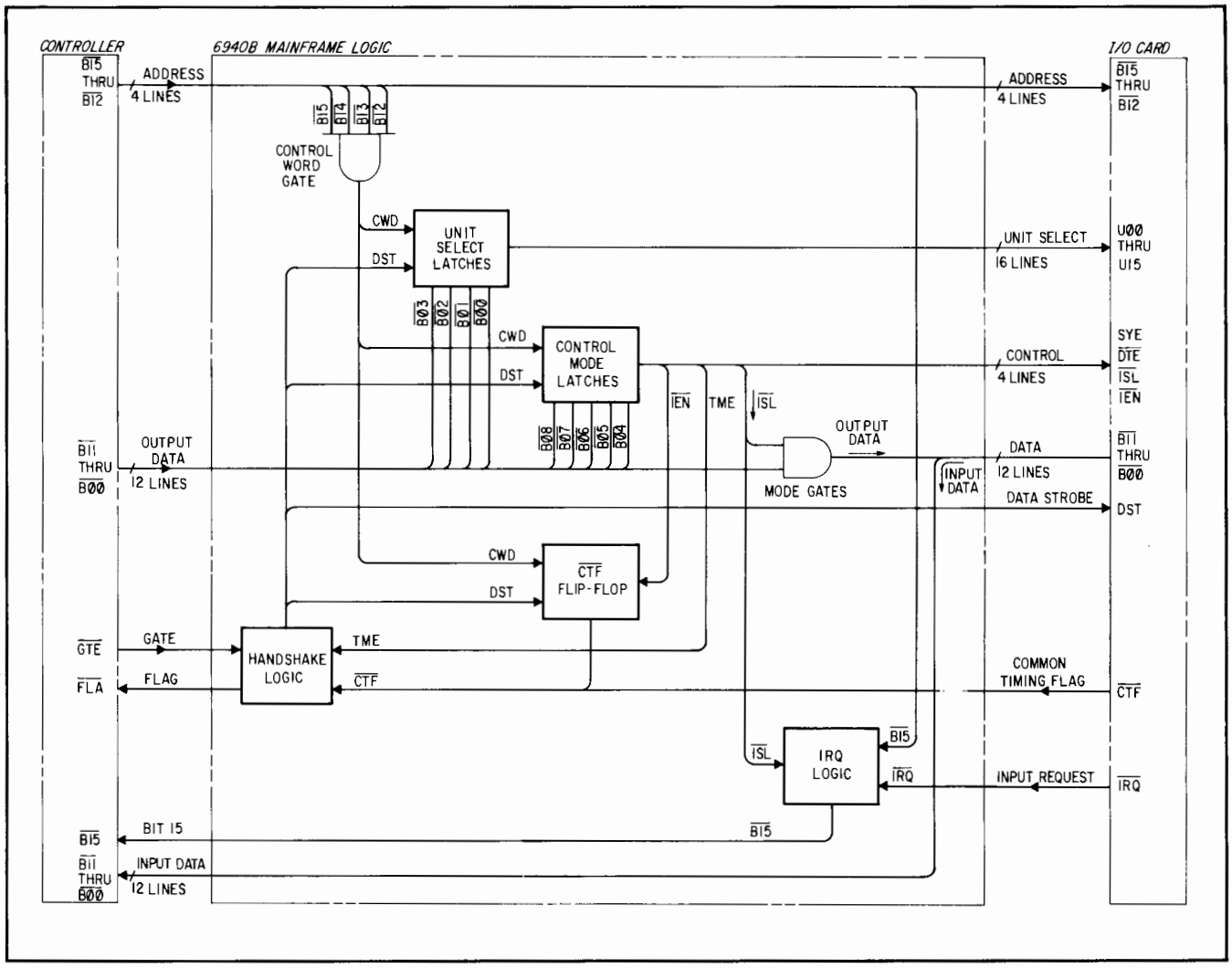


Figure 2-3. Multiprogrammer 6940B, Block Diagram

Table 2-1. Slot Address Codes

ADDRESS	$\overline{B15}$	$\overline{B14}$	$\overline{B13}$	$\overline{B12}$
Slot ØØ	Ø	Ø	Ø	Ø
Slot Ø1	Ø	Ø	Ø	1
Slot Ø2	Ø	Ø	1	Ø
Slot Ø3	Ø	Ø	1	1
Slot Ø4	Ø	1	Ø	Ø
Slot Ø5	Ø	1	Ø	1
Slot Ø6	Ø	1	1	Ø
Slot Ø7	Ø	1	1	1
Slot Ø8	1	Ø	Ø	Ø
Slot Ø9	1	Ø	Ø	1
Slot 10	1	Ø	1	Ø
Slot 11	1	Ø	1	1
Slot 12	1	1	Ø	Ø
Slot 13	1	1	Ø	1
Slot 14	1	1	1	Ø
Control Word	1	1	1	1

the format of the data word; that is, the address and data portions of the word are processed separately by the mainframe logic.

2-19 As shown in Figure 2-3 the data signal lines are distributed to the unit select latches ($\overline{B00}$ through $\overline{B03}$), the control mode latches ($\overline{B04}$ through $\overline{B08}$), and the mode gates (all twelve bits) which, depending on the mode of operation, either terminate the signals or transmit them in parallel to every card slot in the system. Data signals sent to the unit select and control mode latches are ignored except when a control word is programmed and detected by the control word gate. As was previously explained, a control word is specified by the presence of logical "1's" on all four address lines. Therefore, a control word is a special case of the data word, and has the format shown in Figure 2-5.

2-20 The address signal lines are sent to the control word gate and to each of the I/O card slots where a

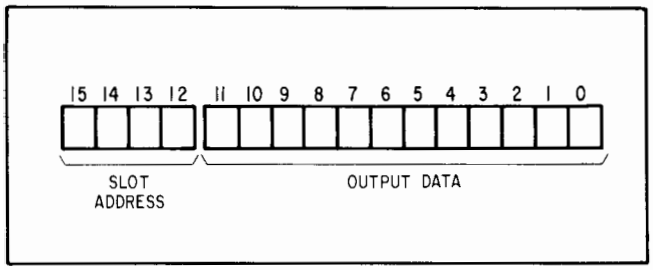


Figure 2-4. Data Word Format

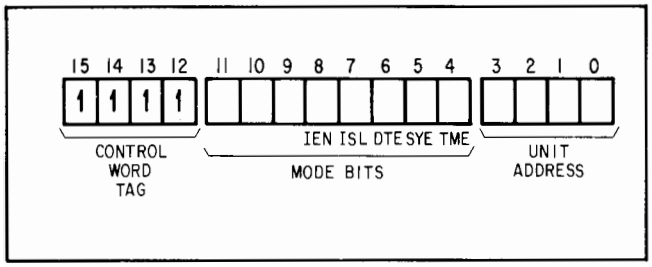


Figure 2-5. Control Word Format

combination of hardwired logic in the mainframe and address decode gates (similar to the control word gate) on the cards permit the addressed card, and only the addressed card, to be partially enabled. A card is completely enabled when, (1) the card slot address sent by the controller matches the address of the slot in which the card is located, (2) the unit address is similarly correct, and (3) the appropriate control signals are received by the card. One additional signal (data strobe or DST) is required whenever data is transferred from the controller to the multiprogrammer. Depending on the card model and the system operating mode, the start of the data strobe initiates one of several possible responses to data present on lines $\overline{B00}$ through $\overline{B11}$.

2-21 Basic Data Transfer Cycle

2-22 Data strobe is a function of the gate signal sent by the controller and is developed by the handshake logic. Referring back to Figure 2-1, the most basic form of the data transfer cycle takes place as follows:

1. The controller transfers data to its output register, placing that data on the sixteen lines (four address and twelve data) that connect to the 6940B input circuitry. To indicate that it wishes to initiate a data transfer, the controller drives the gate ($\overline{GTE}$) line low.

2. About two μsecs after the leading edge of $\overline{GTE}$, handshake logic generates the data strobe. At some point in time after the beginning of DST, the handshake logic returns the leading edge of flag ($\overline{FLA}$), indicating that processing has begun.

3. The controller responds to the leading edge of flag by resetting the gate line to its high state.

4. The handshake logic detects the resetting of the gate signal and terminates DST. When the multiprogrammer has finished processing, the handshake logic returns the trailing edge of flag, thereby concluding the transfer cycle.

2-23 The sequence of events outlined above is typical of all data transfers in the output and control modes. Information transferred in this manner (either to a card or to the unit select and control mode latches) remains stored until the particular address is again specified and a transfer cycle initiated by the controller. Input operations (transfer of data from multiprogrammer to the controller) do not necessarily require use of the gate signal and subsequent handshake cycle. Note also that the sequence outlined above is general enough to describe either the automatic handshake or the extended handshake. Detailed descriptions of both handshake sequences are found in Paragraphs 2-33 through 2-39.

2-24 Control Mode Bits

2-25 Table 2-2 identifies the various control mode bits and summarizes their effect on the operation of the multiprogrammer system. Control signals are distributed throughout the entire system and are totally independent of the slot and unit select address functions. The action of the unit select signals has been explained previously and will not be discussed again here.

2-26 System Enable (SYE). The SYE signal effects the operation of output cards only. With the exception of a few cards (unaffected by SYE), the SYE signal provides a system wide control of output cards which acts independently of the address function; in other words, the SYE signal influences the operation of cards regardless of whether or not they are addressed. When SYE is on, output cards are enabled to transmit any data currently latched in local (on card) storage registers. If SYE is off, data connections between output cards and the external environment are open-circuited or otherwise disabled. The SYE circuitry on individual cards has no control over the address and storage functions, meaning that card outputs may be inhibited without effecting the controller's ability to address and load data into local registers. If SYE is programmed on after a number of cards have been loaded with data in this manner, the cards will simultaneously output that data to external devices when the control word is gated.

2-27 Preset circuitry in the 6940B mainframe ensures that SYE is held off when the system is first powered up. Additional preset circuitry on the individual cards disables outputs (even if SYE is programmed on) until the card is addressed and strobed for the first time. Also, as explained previously, the SYE signal is used to protect against possible damage in the event that a cable is accidently disconnected.

Table 2-2. Control Bits

BIT POSITION	CONTROL BIT	FUNCTION
4	TME (Timing Mode Enable)	Modifies action of the handshake logic by suppressing the automatic flag and causing the $\overline{FLA}$ signal to follow the state of the $\overline{CTF}$ (common timing flag) signal.
5	SYE (System Enable)	Absence of this signal automatically deactivates certain cards to a "safe" state.
6	$\overline{DTE}$ (Data Transfer Enable)	Controls data transfer and timing signals (extended handshake) on certain output cards.
7	$\overline{ISL}$ (Input Select)	Terminates output data in the mode gates, thereby permitting input data to be returned to the controller from an addressed input card.
8	$\overline{IEN}$ (Interrupt Enable)	Used in conjunction with TME to modify the action of the handshake logic during input operations.

If the connection is broken in the middle of a chain of mainframes, all units downstream from the break are disabled. Cards within these units will retain previously programmed data, however, and will output that data once the connection is restored. If a mainframe in the midst of a chain is powered down, the SYE signal will again be disabled, inhibiting outputs from that unit and from any units downstream. Once power is restored, all outputs will resume as previously programmed except in the unit that was turned off. Cards in that mainframe will be preset, requiring that they be individually addressed with data and strobed before the system is fully returned to the state that existed just prior to the power off condition. In all cases, absence of the system enable signal serves to prevent uncontrolled outputs, thus reducing the possibility of stimulating external devices with undesired contact closures, excessive voltages, etc.

2-28 Data Transfer Enable ($\overline{DTE}$). The $\overline{DTE}$ signal provides a mechanism for synchronizing outputs from several cards. The $\overline{DTE}$ signal influences the operation of four card models, all of which function as output devices. Two of these, the voltage and current D/A cards, have dual rank storage registers as shown in Figure 2-6A. The first register accepts data from the mainframe when the card is addressed and gated with the DST signal. The second register receives data from the first and outputs it to the D/A converter, but does not accept this new data unless $\overline{DTE}$ is programmed on. Since the converter is connected directly to the external device (if SYE is on), outputs from several D/A cards can be held constant while new data is loaded into first rank storage on each of the cards. $\overline{DTE}$ can then be programmed on, simultaneously transferring data on each of the cards from first to second rank storage, thereby changing the outputs of all such cards in unison. This feature can be defeated by programming $\overline{DTE}$ on prior to loading data into the first register. Data sent and gated under these circumstances immediately passes through first rank storage to the second rank and then to the converter.

2-29 The two other cards effected by $\overline{DTE}$ do not have dual rank storage, but instead have a gate/flag interface with the external environment, (see Figure 2-6B). These cards inhibit this gate signal unless $\overline{DTE}$ is on. The controller may sequentially address cards and load data into local storage with the DST signal. If SYE is on, the data will be transferred to the external device, but the gate signal to the external device will be suppressed until $\overline{DTE}$ is programmed on and latched with DST, at which time all loaded cards will simultaneously generate gate signals. If $\overline{DTE}$ is programmed on first, each card will generate its gate signal when addressed and strobed. The gate signals may be used to enable storage circuits in the external device. In this case, the combination of the external storage circuits and the storage circuits on the cards provides a dual rank storage system similar to that described in Paragraph 2-28.

2-30 Input Select ($\overline{ISL}$). The $\overline{ISL}$ signal causes the most fundamental change in system operating mode. By terminating output data in the mode gates, the $\overline{ISL}$ signal effectively converts the mainframe from an output multiplexer to an input multiplexer. The address portion of the output word remains functional and when combined with $\overline{ISL}$ on an input card, enables the addressed card to return data to the

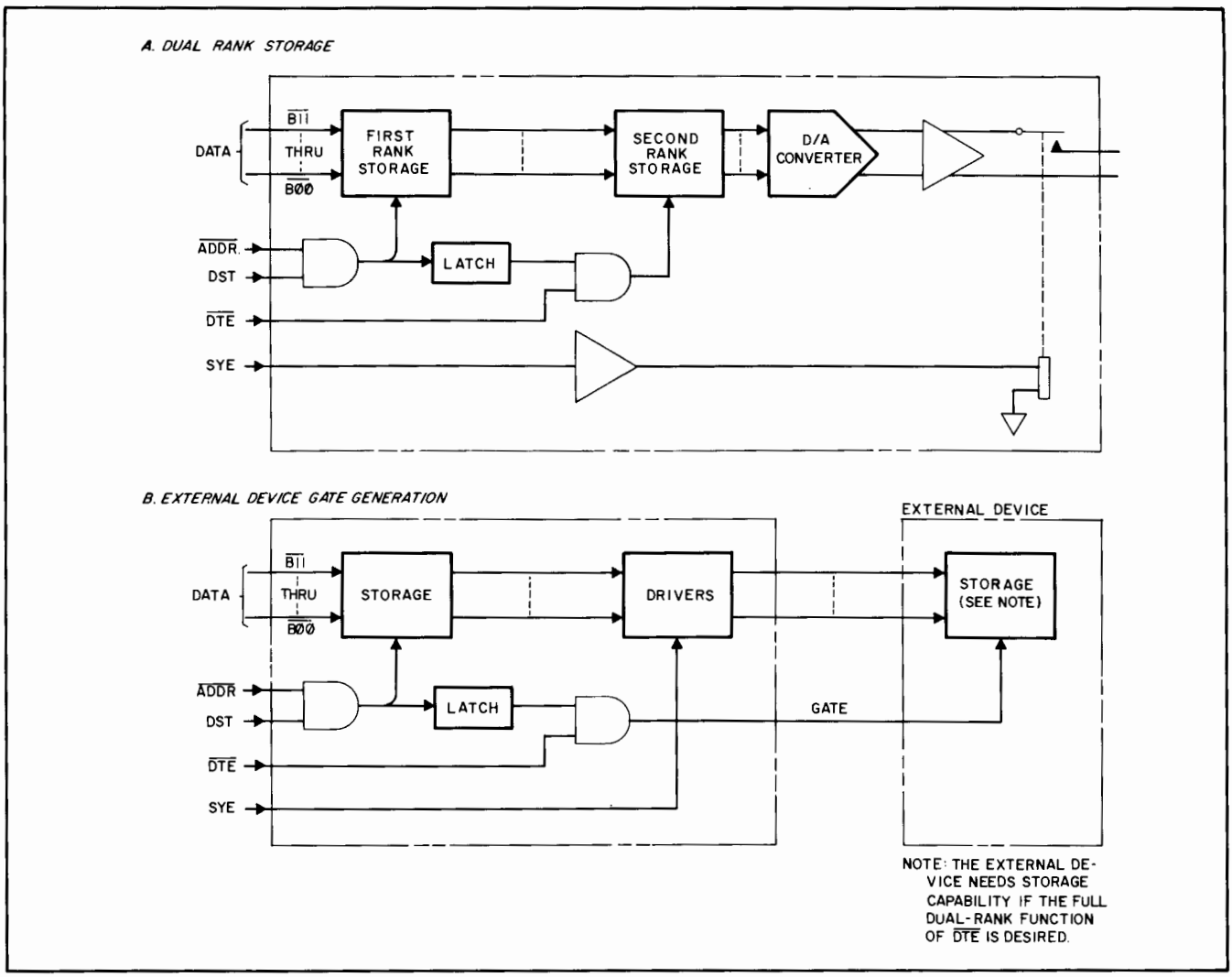


Figure 2-6. Data Transfer Enable (DTE) Circuits

controller. The output word sent by the controller when the system is in the input mode is called an address word and has the format shown in Figure 2-7.

2-31 Figure 2-8 shows the format of the return data word. The twelve data bit lines (B00 through B11) follow the state of the corresponding output data bits so long as ISL is off; when ISL is on, they are driven by the addressed card. Bits B12 through B14 are not used in the return data word; but B15 is, and functions in a manner similar to the twelve data bits. Referring back to Figure 2-3, the IRQ (Input Request) logic selects the source of the B15 signal according to the state of ISL. If ISL is off, B15 of the output word is returned to the controller. If ISL is on, however, B15 of the return data word no longer follows the state of the output bit, but instead reflects the state of the IRQ signal line. The IRQ line is returned from every I/O slot in the system and may be driven by the currently addressed card provided that card is one of the models equipped with IRQ drive circuitry. This signal indicates whether or not the addressed card has completed a data transfer cycle with the external device and is used as the basis for a status polling routine which is part of the interrupt input mode programming sequence (see Paragraphs 2-52 through 2-59).

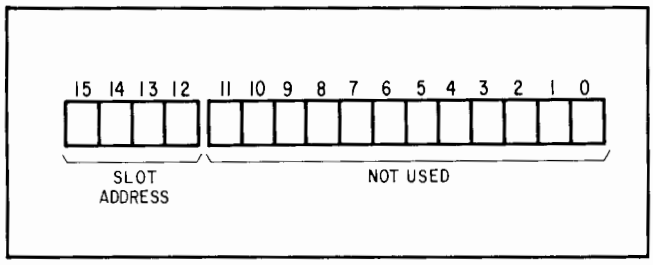


Figure 2-7. Address Word Format

2-32 Timing Mode Enable (TME) and Interrupt Enable (IEN). Up to this point, explanations of the handshake cycle have stressed the sequence of events without much reference to the factors controlling those events.

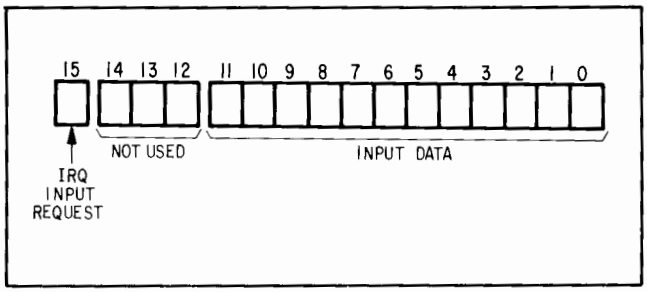


Figure 2-8. Return Data Word Format

An expanded discussion of these factors is necessary if operation of the TME (Timing Mode Enable) and the $\overline{IEN}$ (Interrupt Enable) control signals is to be understood. Figure 2-9 shows a portion of the functional block diagram given in Figure 2-3. Only those elements effecting the handshake are shown, and have been slightly re-arranged in order to clarify the following discussion.

2-33 Automatic Handshake. The TME signal determines whether the multiprogrammer operates in the automatic or extended handshake mode. When TME is off (automatic handshake), the handshake logic is the source of the flag signal returned to the controller. The DST signal, which is generated by the handshake logic in response to the controller's gate ($\overline{GTE}$), causes the handshake logic to return the leading edge of flag ($\overline{FLA}$). When the controller resets $\overline{GTE}$, DST is terminated and the handshake logic returns the trailing edge of flag, completing the transfer cycle. In this mode of operation both edges of $\overline{FLA}$ are an automatic response to the controller's gate. The handshake logic has absolute control of the flag signal, and together with the controller forms the complete circuit necessary to synchronize a data transfer.

2-34 Extended Handshake. When TME is on, the automatic handshake just described is suppressed, and the circuit elements shown to the right of the handshake logic in Figure 2-9 assume control of the flag signal. The data strobe (DST) and the gate signal sent to the external device ($\overline{GAT}$) act as extensions of the controller's gate ($\overline{GTE}$). In a corresponding manner, the flag returned from the external device ($\overline{FLG}$) and the common timing flag ($\overline{CTF}$) become the source of the flag returned to the controller ($\overline{FLA}$). These six signals, then, are used to implement the extended handshake. Since a number of card models do not have a gate/flag interface with the external device, the complete circuit is not always used; the $\overline{FLA}$ signal, however, always follows the state of the common timing flag when TME is on.

2-35 The $\overline{CTF}$ signal line is returned to the handshake logic from the $\overline{CTF}$ flip-flop and from all I/O slots in the system. The entire circuit is arranged in a wire-OR'ed configuration such that any single source (a card or the $\overline{CTF}$ flip-flop) can drive the $\overline{CTF}$ line to a low state. It is not unusual for $\overline{CTF}$ to be driven by several sources at the same time, in which case it will be held low as long as it is driven by at least one source. As long as $\overline{CTF}$ is low, $\overline{FLA}$ will also be held low. The controller will interpret the low state of the $\overline{FLA}$ signal as an indication that the multiprogrammer is processing data and therefore is "busy".

2-36 Except in the unique situation when the $\overline{IEN}$ control signal is used to arm input cards that have the W6 jumper option installed, (arming operations are explained in Paragraphs 2-52 through 2-54), no source can drive $\overline{CTF}$ unless it has been addressed and gated with DST. In most cases, additional conditions (which vary from one card model to another) must be satisfied before the source is fully activated. Since the $\overline{FLA}$ signal returned to the controller is dependent upon the state of the $\overline{CTF}$ line when the multiprogrammer is in the extended handshake mode, care must be taken to ensure that one or more sources are or can be activated whenever the controller gates the system.

2-37 The $\overline{CTF}$ flip-flop is always activated when a control word in which $\overline{IEN}$ is off is programmed and gated. Its only function is to guarantee that a flag is returned to the controller at the time a control word with TME on is gated

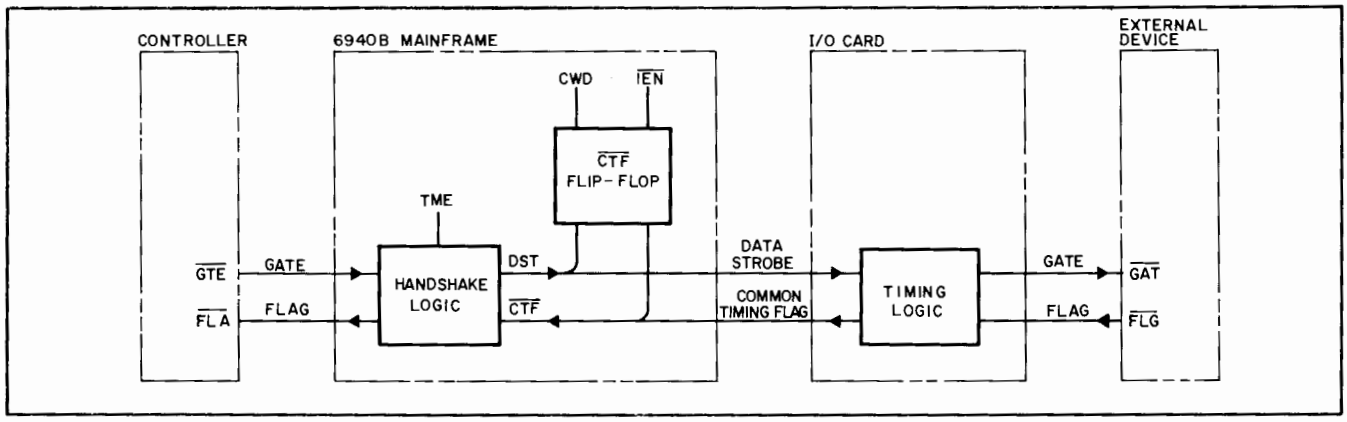


Figure 2-9. Extended Handshake Circuitry

and either of two situations exists:

1. No card has previously been activated to drive $\overline{CTF}$; or
2. All previously activated cards have completed their transfer cycles and have returned to a de-activated state.

2-38 When a control word with $\overline{IEN}$ on is gated, the action of the $\overline{CTF}$ flip-flop is suppressed. This control state is established as part of a specific sequence used during the interrupt input mode of operation and should not be programmed except as part of that sequence. The interrupt mode is described in Paragraphs 2-52 through 2-59.

2-39 Table 2-3 lists the cards with $\overline{CTF}$ drive circuitry and indicates which control signals must be present for the card to be fully activated. Cards not included have no means for driving $\overline{CTF}$ and must not be addressed and gated when $\overline{TME}$ is on since a flag signal will not be returned to the controller.

2-40 PROGRAMMING SEQUENCES

2-41 Although the exact order in which control modes are established and cards activated depends in part upon the particular application, certain generalized programming sequences can be identified and explained. Figure 2-10 shows the sequences applicable to the basic classes of operation in the TME mode. These classes may be distinguished from one another by the answers to three questions:

1. Is the transaction an input or an output operation?
2. Is one card used in the transaction or are several cards used?
3. If several cards are used, do complete transactions occur sequentially (serially) or do they occur concurrently (in parallel)?

2-42 For the purposes of the following explanations, it is assumed that SYE has been programmed on previously and that no cards are activated when the programming sequence is started. Note that the source of the $\overline{CTF}$ signal used to control $\overline{FLA}$ is given in brackets whenever Figure 2-10 indicates that a gate is sent by the controller. Timing diagrams showing the transitions of the $\overline{GTE}$ and $\overline{FLA}$ signals in both the automatic and the extended handshake modes are provided in Figure 2-11. Arrows indicate that a given transition automatically brings about the one immediately following it.

2-43 Output Operations (Figure 2-10)

2-44 **Single Transaction.** An output transaction with a single card is the least complicated data transfer operation. The output mode is first established by programming and gating a control word with $\overline{DTE}$ and TME on. $\overline{ISL}$ and $\overline{IEN}$ are off. $\overline{DTE}$ need not be on if the particular card is not controlled by it, but no harm is done if it is on, and it is generally good practice to program TME and $\overline{DTE}$ on together during output operations since many output cards

Table 2-3. $\overline{CTF}$ Activation Requirements

MODEL	DESCRIPTION	DST	$\overline{DTE}$	$\overline{ISL}$	$\overline{IEN}$
69321B	Voltage D/A	Yes	Yes	—	—
69325A	Power Supply	Yes	—	—	—
69326A	Amplifier	Yes	—	—	—
69327A	Programming	Yes	—	—	—
69328A	Cards	Yes	—	—	—
69330A	Relay Output	Yes	Yes	—	—
69331A	TTL Output	Yes	Yes	—	—
69335A	Stepper Motor	Yes	—	—	—
69370A	Current D/A	Yes	Yes	—	—
69421A	Voltage Monitor	Yes	—	—	—
69431A	Digital Input	Yes	—	See Note	
69433A	Relay Readback	Yes	—	—	—
69434A	Event Sense	Yes	—	See Note	
69436A	Process Interrupt	Yes	—	See Note	
69500A thru 69513A	Resistance Programming	Yes	—	—	—
69600A	Timer	Yes	—	See Note	

Note: The $\overline{ISL}$ signal is sufficient to activate these cards. If W6 is installed, $\overline{IEN}$ is sufficient.

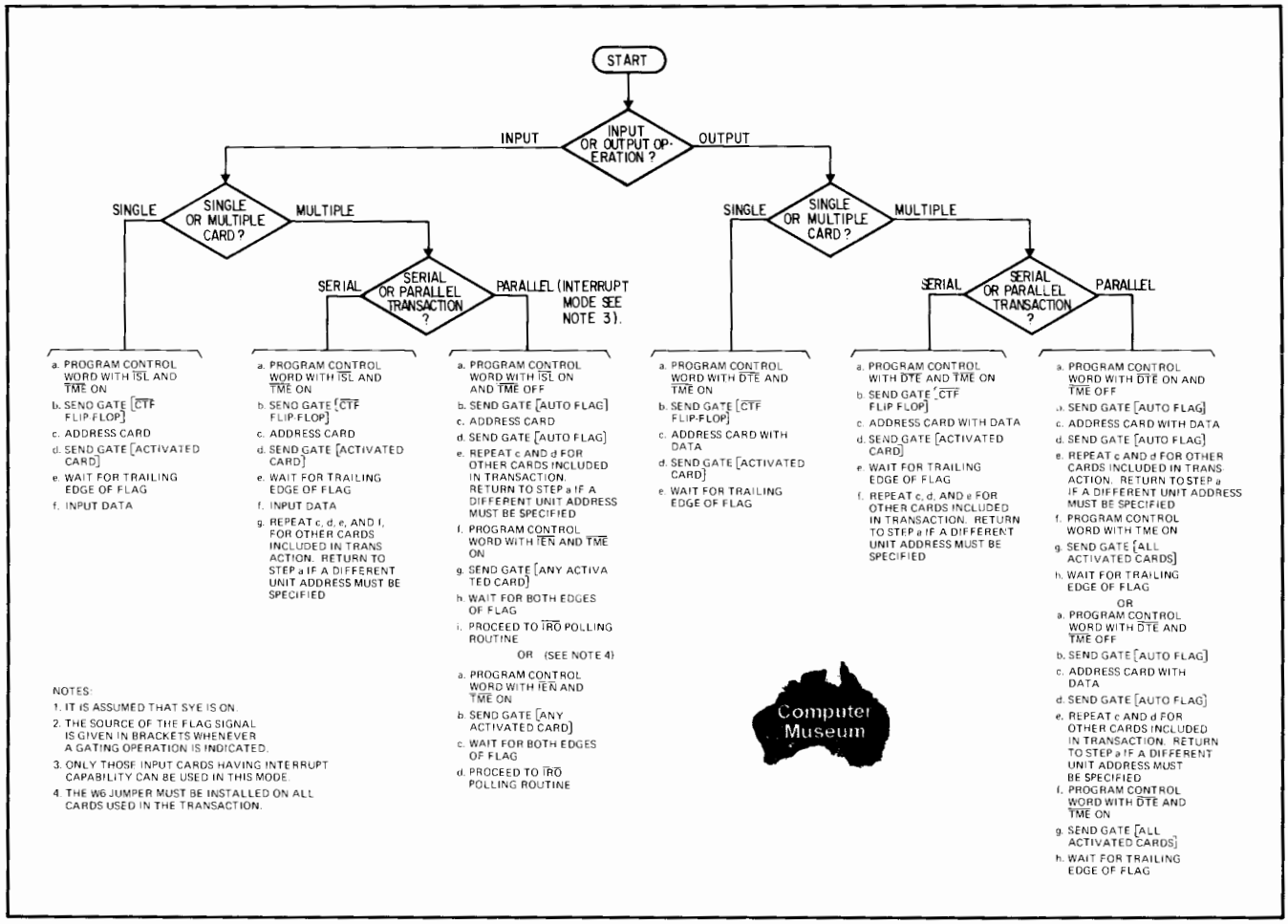


Figure 2-10. Programming Sequences in the Timing Mode

cannot drive $\overline{CTF}$ unless $\overline{DTE}$ is on. The card is then addressed and gated, and immediately returns the leading edge of $\overline{CTF}$. Depending on the card model, the trailing edge of $\overline{CTF}$ is controlled by circuits on the card itself, or by the external device. In either case, the transaction is concluded when the trailing edge of $\overline{CTF}$ is detected by the handshake logic and returned to the controller as the trailing edge of $\overline{FLA}$.

2-45 Serial Transactions. Serial transactions with output cards may be programmed by simply repeating the sequence just described for each card in turn. The serial method is disadvantageous when the number of cards is large and the periods of the successive handshake cycles are long. The controller must wait for each card to return the trailing edge of flag before it can proceed to the next card, with the result that the total operation consumes an unnecessarily large amount of time and makes inefficient use of the controller's processing capability. Consequently, the serial method should not be used unless a strict sequential operation is required by a particular application.

2-46 Parallel Transactions. The two programming sequences for parallel output transactions shown in Figure 2-10 solve this problem by not programming TME on until all of the cards have been activated in the automatic handshake mode. This high speed approach is possible since TME has no effect on the activation of cards and their subsequent control of $\overline{CTF}$, but only determines whether or not the $\overline{FLA}$ signal returned to the controller by the handshake logic will follow the state of the common timing flag. Once all cards have been activated, a control word with TME on is programmed and gated. The $\overline{CTF}$ flip-flop returns the leading edge of flag when the control word is gated, but the wire-OR'ed nature of the $\overline{CTF}$ circuitry will suppress the trailing edge until all of the activated cards have "timed out".

2-47 The two parallel output sequences differ from one another in the manner in which $\overline{DTE}$ is used. The first programming sequence is somewhat faster in execution because each card begins its transfer cycle as soon as it is addressed and gated. Cards with long transfer cycles can be activated early in the sequence, while cards with short cycles can be activated last, thus minimizing the time used for the complete transaction. The second sequence

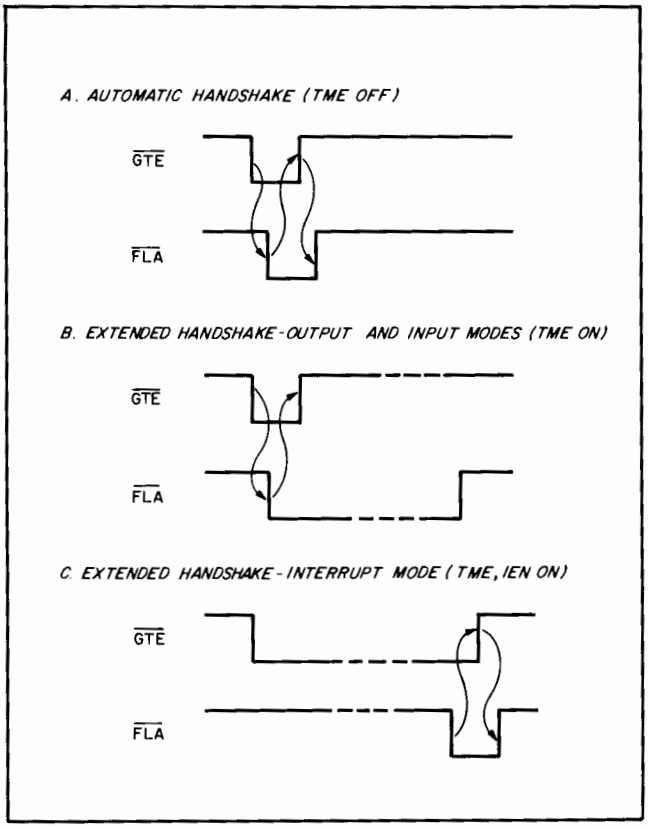


Figure 2-11. Handshake Timing

sacrifices speed in order to provide full parallel output capability. This method uses only those card models controlled by the $\overline{DTE}$ signal and is identical to the first except that neither TME nor $\overline{DTE}$ is programmed on until all cards have been addressed and gated. Since $\overline{DTE}$ must be on for these card models to be fully activated, they are not able to drive the $\overline{CTF}$ line, but do remain in a partially activated state even when they are no longer addressed. When the control word with TME and $\overline{DTE}$ on is programmed and gated, all cards are simultaneously enabled to output new data and fully activated to drive the $\overline{CTF}$ line.

2-48 If TME is not programmed on, all output cards (including those without $\overline{CTF}$ drive circuitry) may be operated in any of the four programming sequences discussed so far. The distinction between the serial sequence and the first parallel sequence is no longer significant, however, since the flag is now returned automatically by the handshake logic. The second parallel sequence is different only because it makes use of the $\overline{DTE}$ function to synchronize outputs to the external environment.

2-49 Input Operations (Figure 2-10)

2-50 Certain input cards may also be operated in the output mode, and are generally programmed using the single or serial sequences with TME off. Just as there are input cards which may be used in the output mode, so there are output cards which have a secondary function that is used in the input mode.

NOTE

Unlike output operations where a gate signal is always required to strobe data into local storage registers, use of the gate depends on the particular card model during input transactions.

2-51 Cards with input capability have circuitry for driving either the $\overline{IRQ}$ signal line, some or all of the twelve return data lines, or for driving all thirteen lines. Any card in this group is enabled to drive the return data bus if it is addressed with $\overline{ISL}$ on, regardless of whether or not it is gated. Cards without $\overline{CTF}$ circuitry must not be gated if TME is also on, and are usually operated without a gate in the automatic handshake mode where TME is off and the possibility of a system "hang-up" is eliminated. The remaining cards with input capability may be used in this mode or in one or more of the input programming sequences given in Figure 2-10.

2-52 **Parallel (Interrupt Mode) Transactions.** Cards which function in the parallel input mode (interrupt mode) are the only cards able to drive the $\overline{IRQ}$ signal line. These cards also have $\overline{CTF}$ drive circuitry and, with the exception of the 69431A digital input card, respond to and act upon data when addressed and gated in the output mode. Only one card (the 69600A timer card) actually transmits data to the external environment, however, the others use data received in the output mode to modify the manner in which they react to input signals. All cards with interrupt capability must be armed before they are enabled to drive either the $\overline{IRQ}$ or the $\overline{CTF}$ signal lines in the interrupt mode.

2-53 Individual arming is accomplished by addressing and gating the card when the system is in the input mode (a control word with $\overline{ISL}$ on must have been programmed and gated previously). Group arming, on the other hand, is made possible by a hardware option available on all cards with $\overline{IRQ}$ circuitry. A jumper (W6), when installed, permits the controller to arm cards by programming the interrupt mode (a control word with $\overline{IEN}$ and TME on is programmed and gated). As they are armed, the cards are enabled to begin a data transfer cycle with the external device and to drive the $\overline{IRQ}$ signal line.

2-54 As shown in Figure 2-10, steps (a) through (g) given in the first parallel input programming sequence and steps (a) and (b) given in the second arm the cards and place the system in the interrupt mode. Since the $\overline{IEN}$ control signal suppresses the $\overline{CTF}$ flip-flop and TME suppresses the

automatic handshake, the controller depends on the armed cards to return the leading edge of flag. When operated in the interrupt mode, cards do not return a leading edge of flag until they have completed a data transfer cycle with the external device. The first card to complete transfer drives $\overline{CTF}$ low and causes the handshake logic to return the leading edge of flag, whereupon the controller resets $\overline{GTE}$ and the handshake logic responds by terminating DST. Once the data strobe is terminated, the $\overline{CTF}$ circuitry on all armed cards is disabled and the trailing edge of flag is returned to the controller indicating that an interrupt has occurred. Unlike the parallel output mode where the trailing edge of flag is controlled by the last card to complete its operation, the interrupt mode flag is controlled by the first card.

2-55 **$\overline{IRQ}$ Poll.** At this point the controller polls all armed cards by sequentially addressing them without a gate in the input mode ($\overline{ISL}$ is on and TME is off) and examining the $\overline{IRQ}$ bit (bit fifteen of the return data word) returned from each card as it is addressed in order to determine which card completed its transfer cycle and generated the interrupt. If the addressed card has completed a transfer cycle, the $\overline{IRQ}$ signal is returned to the controller low (true), otherwise it is returned high (false). Depending on the card model, a high state on the $\overline{IRQ}$ line simply means that the data transaction is still in process, in other cases it indicates that the addressed card has not detected a change of state in the external device. Since it is possible that several cards have completed their transfer cycles, the order in which cards are polled establishes an interrupt priority system in software.

2-56 When an interrupting card is found, the controller may process the return data word according to the demands of the particular program. Once this is completed, the controller must either recycle or disarm the interrupting card since it will generate an interrupt as soon as the system is again gated with $\overline{IEN}$ on unless it is reset (the requirements for interrupting, recycling and disarming are different for each card model and are explained in detail in Chapter VI). If a card is recycled, it remains armed but will not generate an interrupt until it has completed another transfer cycle; if it is disarmed, it becomes inoperative in the interrupt mode until it is again armed. Recycling or disarming should be considered part of any gated data input sequence since cards with interrupt capability must be reset in one way or the other whenever they are used, regardless of whether the system is in the interrupt mode or not.

2-57 Once the interrupting card is reset, the controller may continue with the poll or it may re-establish the interrupt mode by gating the system with $\overline{IEN}$ on. If the first interrupt was generated by several cards at the same time or a card completed its transfer cycle while the polling routine was conducted, a new interrupt will be generated immediately;* if not, the controller will again wait for both edges of flag. The controller may also re-establish the interrupt mode, proceed to a totally unrelated operation, and then return to the polling sequence at some later time, either as the interrupt occurs (if the controller has a hardware interrupt system) or when directed to do so by the program.

2-58 The entire interrupt sequence may be terminated after an interrupt has occurred provided that the system has not been gated again with $\overline{IEN}$ on. Even if the gate has been sent, the mode may still be aborted, but the controller's gate must now be reset by a program command. In either case it is a good policy to disarm any active cards.

2-59 The W6 jumper option should be used with caution. All cards in the system which have W6 installed will be armed whenever IEN is programmed on and gated. This situation can create difficulties if the programmer is not fully aware of its implications. For instance: the $\overline{IRQ}$ polling routine must be conducted with $\overline{ISL}$ on and TME off, but $\overline{IEN}$ has no influence during this operation and may either be left on or turned off. If $\overline{IEN}$ is turned off, however, and the interrupting card is disarmed, the interrupt mode cannot be programmed back on without re-arming the card. The problem may be solved by leaving $\overline{IEN}$ on during the polling and disarming operation. Similarly, cards with W6 installed may not be individually armed, but must instead be selectively disarmed after $\overline{IEN}$ is programmed on.

2-60 Single and Serial Transactions. Returning to Figure 2-10, it may be seen that single card and serial input transactions are related in the same sense that those two classes of output transactions are; the serial process is nothing more than a repetition of the single card operation. Assuming that all cards involved are able to drive the $\overline{CTF}$ line, the input and extended handshake modes are established by programming and gating a control word with $\overline{ISL}$ and TME on. The first card is then addressed, gated, and immediately drives $\overline{CTF}$ low, returning the leading edge of flag. Once the data transfer between the external device and the card is complete, the card stops driving $\overline{CTF}$ and the trailing edge of flag is detected and returned to the controller by the handshake logic. Thus informed that the data transfer has taken place, the controller maintains the card address and reads back the input word on the return data bus. Card models having the necessary drive circuitry will also return $\overline{IRQ}$ while addressed, but the controller need not test that bit since the trailing edge of flag is only returned when the data transfer is complete and therefore

* The 69434A event sense card is an exception since it cannot detect or store an event unless both DST and $\overline{IEN}$ are on.

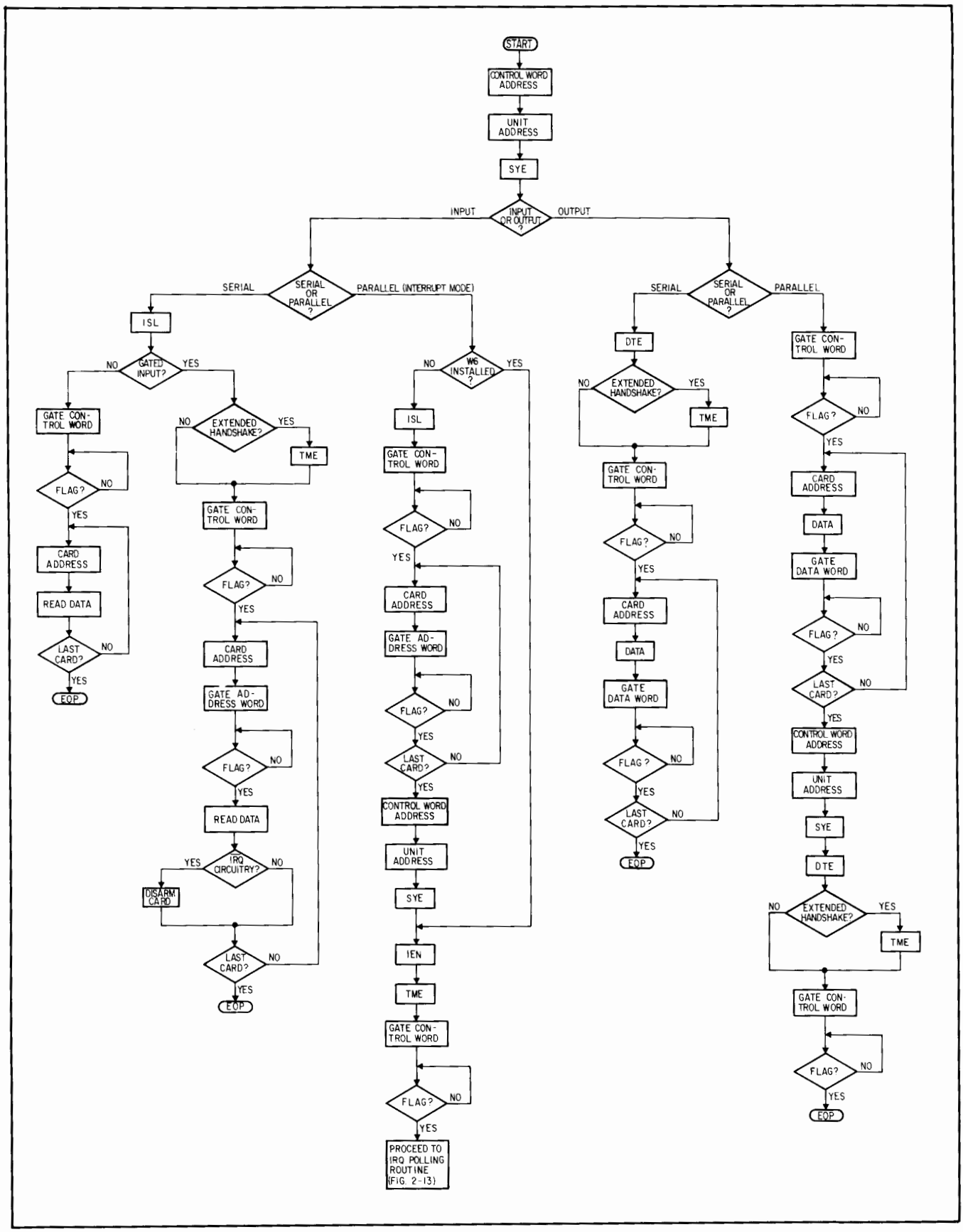


Figure 2-12. Programming Sequences, Flow Chart

is synonomous with $\overline{IRQ}$ true. Regardless of whether the total operation involves one card or many, transactions are completed with a single card at a time and the $\overline{IRQ}$ function is not used. As previously indicated, cards with $\overline{IRQ}$ circuitry are armed when gated with $\overline{ISL}$ on and should be disarmed once the transaction is completed.

2-61 Programming Sequences Flow Charts

2-62 **Main Programming Sequences.** Figure 2-12 presents, in flow chart form, the programming sequences given in Figure 2-10. The automatic handshake modes are also shown. Note that the relationship between single card and serial transactions is made more apparent by combining the two modes into a single sequence which incorporates a "last card?" decision box. When the transaction involves a single card only, the "yes" condition is satisfied on the first pass through the sequence and the flow proceeds directly to the end of operation (EOP). The "no" condition is satisfied during serial transactions until the last card has been programmed.

2-63 The "last card?" decision box also implies the possibility of a different unit address. In such cases, the flow must return to "start" and proceed from there back to the same branch sequence after a control word with the new unit address and appropriate control bits is programmed and gated.

2-64 A "flag?" decision box is shown immediately after each gating operation. The trailing edge of flag is required to satisfy the "yes" condition; the "no" condition is thus sustained until the handshake is complete and provides a "wait for flag" loop.

2-65 **IRQ Poll Sequence.** As shown in Figure 2-12, the interrupt mode programming sequence does not conclude with an end of operation box, but proceeds to the $\overline{IRQ}$ polling routine. A flow chart for that sequence is given in Figure 2-13, and is arranged as a self-contained subroutine.

2-66 FRONT PANEL CONTROLS

2-67 Local Operation.

2-68 The 6940B mainframe is provided with a front panel switch register which is enabled when the data source switch is set to the local position. The switches (with the exception of the data source and power switches) are proximity types actuated by body capacitance and have integral indicator lamps. These lamps monitor the state of the various signal lines (the lamp is on when the signal is in the true state) in both the local and remote operating modes.

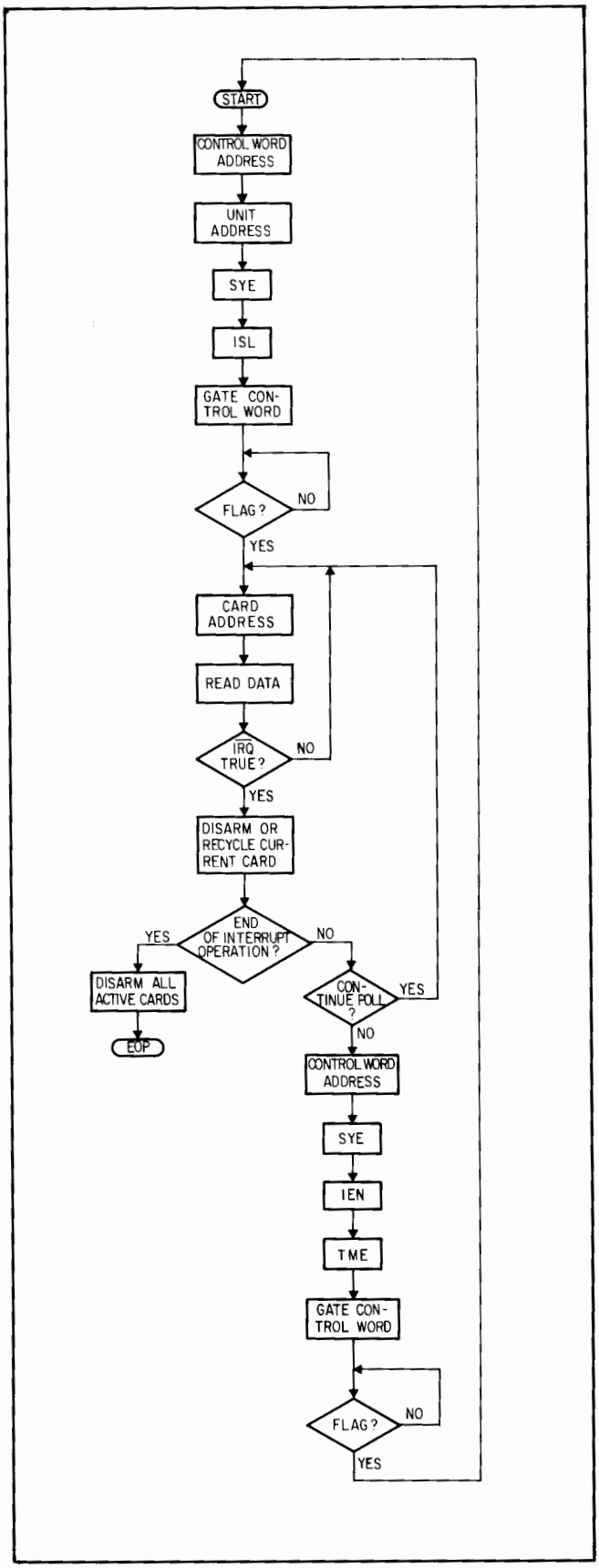


Figure 2-13. IRQ Polling Sequence, Flow Chart

2-69 The sixteen switch/indicators in the top row monitor and/or drive the data signal lines that connect between the multiprogrammer and the controller. If the mainframe is in the output mode ($\overline{ISL}$ is off), the switch register displays the data word either as sent by the controller or as manually programmed on the switches (assuming that the data source switch is set to the local mode). If the input mode is operative ($\overline{ISL}$ is on and a control word address is not specified), the display is a combination of the address word and the return data word. The switches corresponding to bits $\overline{B00}$ through $\overline{B11}$ are not enabled when local operation is selected in the input mode, but the switches for bits $\overline{B12}$ through $\overline{B15}$ are enabled and may be used to specify the card address.

2-70 Return Data Circuit

2-71 The $\overline{IRQ}$ signal is not displayed as bit $\overline{B15}$ on the switch register, but is instead displayed on the return data indicator. The return data switch/indicator also displays and/or drives the $\overline{FLA}$ signal line. Figure 2-14 shows the circuitry used to implement these functions. The return data indicator displays the state of $\overline{FLA}$ regardless of the system operating mode (the lamp is on when $\overline{FLA}$ is low). It will also display the state of the $\overline{IRQ}$ line if $\overline{ISL}$ is also on (the lamp is on if $\overline{IRQ}$ is true). The switch function is enabled only in the local mode, in which case the handshake logic is prevented from driving the $\overline{FLA}$ line and the return data switch becomes the only source for $\overline{FLA}$. The exact reverse situation is in effect when remote operation is selected.

2-72 The load output indicator monitors the controller's gate (the lamp is on when $\overline{GTE}$ is low). The associated switch substitutes drives the $\overline{GTE}$ line when the local operating mode is selected. The clear register switch returns all data switches to their off state and is provided as an operator convenience.

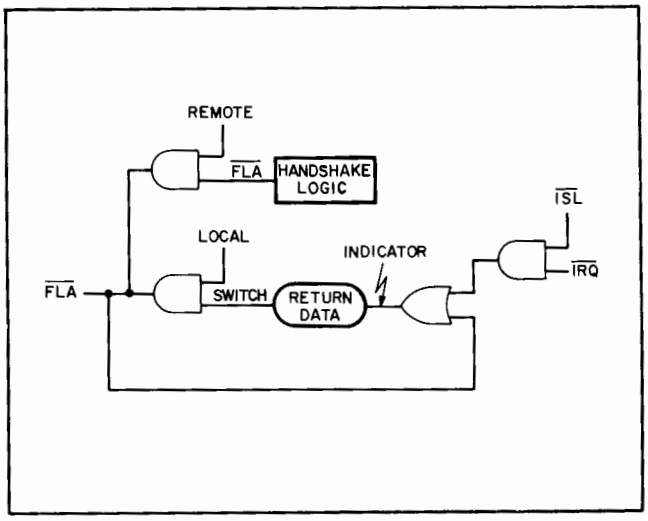


Figure 2-14. Return Data Circuit

Chapter III
MULTIPROGRAMMER INTERFACE 59500A DESCRIPTION

3-1 INTRODUCTION

3-2 The 59500A Multiprogrammer Interface permits bi-directional operation of the 6940B Multiprogrammer and any associated 6941B Multiprogrammer Extenders on the Hewlett-Packard Interface Bus (HP-IB). This chapter provides a basic description of the operations and functions performed by the 59500A and its relationship to the HP-IB and 6940B/6941B.

3-3 Figure 3-1 illustrates the signal flow on the HP-IB between the controller (calculator) and the 59500A Multiprogrammer Interface. The major signal flow between the units (59500A, 6940B, and 6941B) of the multiprogrammer system is also shown.

3-4 Data is transferred over the bus on data input/output lines DIO1-DIO8 utilizing the byte serial technique. Each byte (character) of information, 8-bits in parallel, is transferred onto the bus in serial fashion. Lines DIO1-DIO7 accommodate the 7-bits (1 character) of the ASCII code. Line DIO8 is not used by the multiprogrammer system but can be used by other bus instruments. The state of the attention (ATN) line, controlled by the calculator, determines how the data lines are interpreted. The ATN line is constantly monitored by the 59500A and all other bus devices. When ATN is true, the bus devices (in this case the 59500A) interpret the data on lines as DIO1-DIO8 as instructions (commands) from the calculator. Assume that a command is sent which enables the 59500A to listen in preparation to receive data from a talker (in this case the calculator). Multiprogrammer operations can only be initiated when the 59500A is listening. When the ATN line is false, data is transferred from the calculator to the 59500A. The 59500A converts the serial ASCII characters from the calculator into the 16-bit word format required by the multiprogrammer. Depending upon the data received, the multiprogrammer will perform either an output or an input operation as described in Section II. If the timing mode is not used, the calculator can send the 59500A additional data. However, if it is used, the calculator is prevented from sending the 59500A additional data until the multiprogrammer has completed operations. When an input operation is performed by the multiprogrammer, a return data word is stored in the 59500A. In order for the calculator to receive this data, the calculator must set ATN true, command the 59500A to talk and the calculator to listen. When ATN is false, the 59500A converts the multiprogrammer return

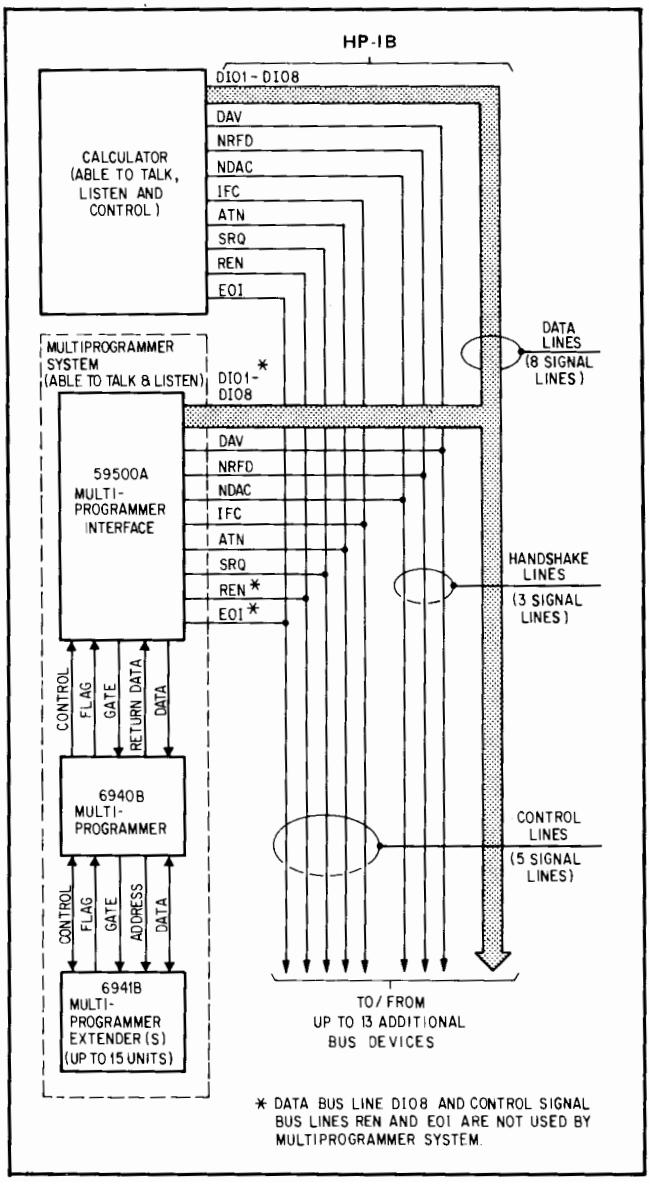


Figure 3-1. HP-IB Connections

data word into serial ASCII characters for processing by the calculator.

3-5 The service request (SRQ) line is used by the 59500A to indicate that the multiprogrammer requires service. The SRQ line is enabled whenever the multiprogrammer is operating in the timing mode and will be set when the multiprogrammer completes an operation(s) or requests an interruption of the current programming

sequence. Control signals from the 6940B to the 59500A implement the service request function. The 59500A also responds to the interface clear (IFC) control signal from the calculator. The IFC signal is used by the calculator to terminate activity on the HP-IB. The 59500A does not respond to the REN (remote enable) and EOI (end or identify) control bus lines.

3-6 A 3-wire handshake process occurs with each character transferred on the HP-IB. A 3-wire handshake cycle takes place when the 59500A is receiving a character and when it is sending a character. The 59500A adapts the 3-wire handshake process used on the HP-IB to the 2-wire method (Gate/Flag) employed by the multiprogrammer. The 3-wire handshake is implemented by the DAV (data valid), NRFD (not ready for data), and NDAC (data not accepted) bus lines.

3-7 BLOCK DIAGRAM DESCRIPTION

3-8 Figure 3-2 illustrates the major circuits and signal flow through the 59500A. All signal abbreviations (i. e. DAV, NDAC, etc.) on the block diagram refer to the true state of the corresponding signal line. The following paragraphs describe operation of the 59500A in the command mode and in the listen and talk modes. The 3-wire hand-

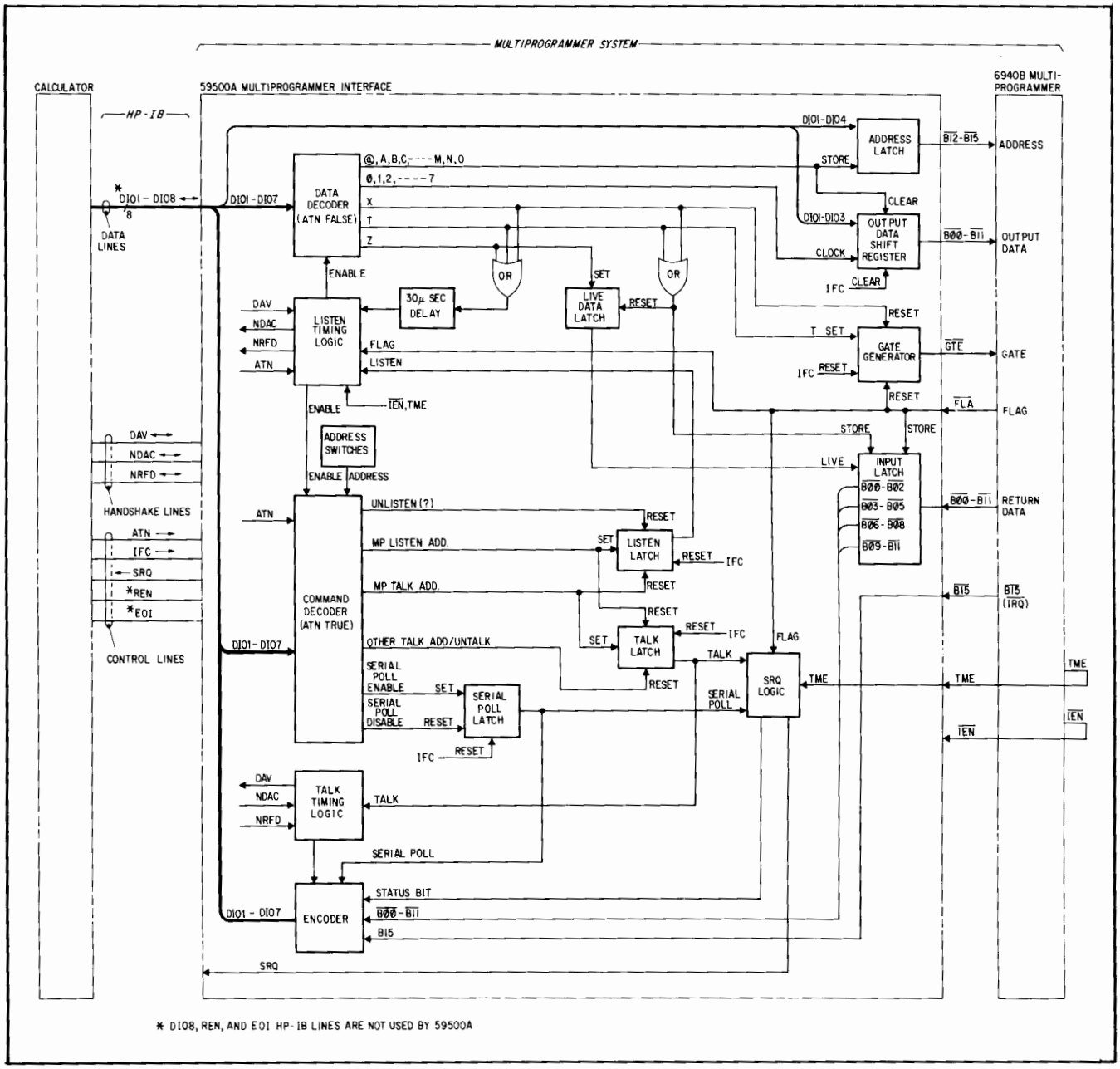


Figure 3-2. Multiprogrammer Interface 59500A, Block Diagram

shake cycle, service request operation and associated serial poll operation are also described.

3-9 Command Mode

3-10 When the ATN line goes true, the command decoder is enabled. The command decoder consists of a talk/listen decoder and a serial poll decoder. The talk/listen decoder receives data lines DIO7 (MSB) through DIO1 (LSB) from the HP-IB and the address code from the address switches (slide switches located on rear of 59500A unit). When the ATN line is true, the talk/listen decoder controls the talk and/or listen latches when any one of the following ASCII characters is decoded:

a. Unlisten command ("?"): Resets the listen latch inhibiting the 59500A from functioning as a listener.

b. Multiprogrammer Listen Address (suggested listen address is "7"): Sets the listen latch and resets the talk latch. The 59500A can function as a listener but not as a talker.

c. Multiprogrammer Talk Address (suggested talk address is "W"): Sets the talk latch and resets the listen latch. The 59500A can function as a talker but not as a listener. The address switches select talk and listen addresses in pairs; a talk address of "W" corresponds to a listen address of "7".

d. Another bus device's talk address or an untalk command ("—"): Resets the talk latch inhibiting the multiprogrammer system from functioning as a talker.

3-11 After one of the above commands or addresses is decoded, the 59500A is enabled to function as a listener or as a talker; or is inhibited from functioning as either. The talk and listen latches will remain either both reset or one set and the other reset until another command is decoded. The talk or listen latch (whichever is set) is reset when the interface clear (IFC) bus line is true. The IFC line is used by the calculator to clear all devices on the HP-IB.

3-12 The serial poll decoder also receives data lines DIO7-DIO1 from the HP-IB. When the ATN line is true, the serial poll decoder controls the serial poll latch if one of the following commands is decoded:

a. Serial Poll Enable: Sets the serial poll latch enabling the 59500A to operate in the serial poll mode.

b. Serial Poll Disable: Resets the serial poll latch inhibiting the 59500A from operating in the serial poll mode.

3-13 Serial polling is a method of sequentially determining which device connected to the HP-IB has requested service. Service request and serial polling operations are described in Paragraphs 3-30 through 3-34.

3-14 A 3-wire handshake cycle occurs with each command character that is transferred from the calculator to every device on the HP-IB. In the command mode (ATN line true), the handshake cycle is automatically implemented by the timing and logic (listen) circuits in the 59500A. The 3-wire handshake process is described in Paragraph 3-35.

3-15 Listen Mode

3-16 When the 59500A listen address has been received (listen latch is set) and the ATN line goes false, the 59500A will interpret alpha and numeric ASCII characters from the HP-IB as data and process them accordingly. The data decoder (5 separate decoders) decodes ASCII characters on data bus lines DIO7-DIO1 as follows:

Letter Decoder — decodes "@" sign and letters from "A" through "O".

Number Decoder — decodes numbers from "0" through "7".

"T", "X", "Z" Decoders — decode gate codes "T", "X", and "Z" respectively.

3-17 When any one of the above characters is received, it is decoded and a handshake cycle between the 59500A and the calculator takes place. Unrecognized characters will be ignored but a handshake cycle between the 59500A and calculator will occur anyway (see Paragraph 3-35).

3-18 Letter Decoder. When the three MSB's (DIO7, 6, 5) specify the ASCII code for an "@" sign or a letter from "A" through "O", the letter decoder stores the remaining four bits (DIO4-DIO1) in the address latch. In addition, a clear signal is generated which presets the output data shift register to zero. The bits stored in the address latch represent a multiprogrammer card slot address or designate a multiprogrammer control word. The bits derived from the ASCII codes for "@" and "A" through "N" represent multiprogrammer card slot addresses 400 through 414, respectively. The bits derived from the ASCII code for "O" designate a multiprogrammer control word. If another "A" through "O" or "@" character is received, it replaces the previous character stored in the address latch.

3-19 Number Decoder. When the first four MSB's (DIO7-DIO4) specify the ASCII code for a number in the range from "0" through "7", the number decoder generates a clock pulse which loads bits DIO3-DIO1 into the bottom three positions of the shift register. As succeeding numbers are received by the number decoder, the data present in the data shift register is shifted up, three bits at a time, as the new data (number) is loaded in. Numbers are shifted in

from the bottom up as they are received. Thus, the first three bits (octal number) received are the most significant output bits in the register. The data shift register holds 12-bits of data, $\overline{B11}$ (MSB) through $\overline{B00}$ (LSB) representing data input to the multiprogrammer. Thus, after four numbers (3-bits each) are received, the data shift register is filled. If less than four numbers are received by the decoder, the most significant bits of the data shift register will be logical zeros because of the prior clearing of the shift register (see Paragraph 3-18). Since some calculators do not transmit leading zeros, this feature of the shift register makes the transmission of leading zeros optional as they will have no effect. If more than four numbers are received, they will continue to be shifted into the data shift register, however, since only 12-bits can be retained, only the last four decoded numbers will be present and all previous numbers will be lost. The shift register is cleared when the IFC line is true.

3-20 "T", "X", and "Z" Decoders. An ASCII "T", "X", or "Z" gate code is decoded on data bus lines DIO7-DIO1 by the applicable decoder. The "T" code, included in all control and data words, generates the multiprogrammer gate signal. The control or data word bits $\overline{B15}$-$\overline{B00}$ (contents of the address latch and output data shift register) are sent to the multiprogrammer along with the gate. The gate signal initiates the multiprogrammer's data storage and/or processing functions specified by the associated control and data words. The "T" code can also be used in an address word to initiate a data strobe to activate the addressed input card, and then store return data bits $\overline{B11}$-$\overline{B00}$ from the addressed input card in the 59500A's input latch. Thus, when the "T" code is used in an address word, it designates a "read with gate" function. The "X" code (read without gate) is used in an address word when it is desired to store data from the addressed input card without gating (strobing) the card. When the "X" code is received, the return data ($\overline{B11}$-$\overline{B00}$) from an addressed input card is stored in the input latch. The "Z" code (read live data) is used in an address word to allow return data ($\overline{B11}$-$\overline{B00}$) from the addressed card to continually pass through the input latch to the encoder. When the "Z" code is received, the input latch is enabled and will remain enabled until a "T" or "X" is received. Thus, the "Z" code allows the contents of the input latch (encoder input) to be continually updated to reflect the return data from the addressed multiprogrammer input card. Control, data, and address words and the uses of the "T", "X", and "Z" gate codes are described in detail in Chapter V.

3-21 The 3-wire handshake cycle for a "T", "X", or "Z" character is delayed 30μsec. When a "T", "X", or "Z" is decoded, the timing logic holds the NRFD bus line true for 30μsec preventing the calculator from sending another character until NRFD goes false. The delay allows the multiprogrammer time to process the data before the calculator sends new data. Note that if a "T" is received, the receipt of another character could be delayed for more than 30μsec since the multiprogrammer flag will override the 30μsec delay (see Paragraph 3-35).

3-22 If a "T" is decoded, the gate generator is set and the live data latch is reset. When set, the gate generator produces the multiprogrammer gate signal which strobes bits $\overline{B15}$-$\overline{B00}$ into the multiprogrammer. The multiprogrammer gate can only be generated when a "T" is decoded. The leading edge of the multiprogrammer flag resets the gate generator and the trailing edge stores the return data bits ($\overline{B11}$-$\overline{B00}$) in the input latch. The flag is also applied to the timing logic (listen) circuit to control the NRFD bus signal. In effect, when a "T" is decoded, the HP-IB 3-wire handshake is converted to the multiprogrammer 2-wire handshake (see Paragraph 3-35).

3-23 If an "X" is decoded, it will reset the gate generator, store data present on return data bit lines ($\overline{B11}$-$\overline{B00}$) in the input latch, and reset the live data latch. The ability to reset the gate generator with an "X" becomes necessary if the gate generator remains set when the multiprogrammer is operating in the interrupt mode. In the interrupt mode, the trailing edge of the flag signal, indicating an interrupt condition, resets the gate generator and activates the service request (SRQ) bus line. If the trailing edge of the flag is not received, the 59500A will "hang up" waiting for an interrupt indication with the gate generator set and the SRQ line false. Consequently, if the interrupt indication (SRQ line goes true) does not occur as anticipated, the calculator can send an "X" to reset the gate generator removing the "hang up" condition. Conditions for setting the SRQ line true are described in Paragraph 3-28.

3-24 When a "Z" is received, the live data latch is set causing the input latch to be continually enabled. For this condition the input latch continually reflects the status of return data bits ($\overline{B11}$-$\overline{B00}$) from the addressed multiprogrammer input card.

3-25 Talk Mode

3-26 Before the 59500A is placed in the talk mode, a control word with a "T" gate code must have previously been received defining the input mode of operation (ISL on). Also, an address word specifying a particular input card's slot address and an accompanying "T", "X", or "Z" gate code must have been previously sent to the 59500A. After the multiprogrammer system is in the input mode (ISL on) and the address word and gate code have been received, the data stored in the 59500A's input latch reflects the data returned ($\overline{B11}$-$\overline{B00}$) from the addressed input card. This information, along with bit $\overline{B15}$, the multiprogrammer $\overline{IRQ}$ bit, is sent to the encoder. At this point, the direction of the data flow on the HP-IB must be reversed by command-

ing the calculator to listen and the 59500A to talk.

3-27 When the 59500A talk address is received (talk latch is set) and the ATN line goes false, the encoder will translate bit 15 into one octal digit and bits $\overline{B11}$-$\overline{B00}$ into four octal digits. Bit $\overline{B15}$ is encoded a 1 if $\overline{IRQ}$ is set and Ø if $\overline{IRQ}$ is not set. The $\overline{IRQ}$ bit is used by certain multiprogrammer input cards to indicate that they contain valid data. IRQ is also used to identify the interrupting card when multiple input cards are used in the interrupt mode. Bits $\overline{B11}$-$\overline{B09}$, $\overline{B08}$-$\overline{B06}$, $\overline{B05}$-$\overline{B03}$, and $\overline{B02}$-$\overline{B00}$ are encoded into ASCII codes for numbers ranging from "1" to "7". Data is transmitted from the encoder onto the HP-IB in the order given above starting with the octal digit representing bit $\overline{B15}$ and ending with the octal digit representing bits $\overline{B02}$-$\overline{B00}$. Carriage return and line feed are added after the last multiprogrammer octal digit ($\overline{B02}$-$\overline{B00}$) as an "end of record" indication to the calculator. As each character is transmitted, a 3-wire handshake occurs between the 59500A and the calculator (see Paragraph 3-35).

3-28 Service Request

3-29 The service request (SRQ) line is used by the 59500A to notify the calculator that the multiprogrammer requires service. Whenever the multiprogrammer is operating in the timing mode (TME on), the SRQ logic circuit in the 59500A is enabled and will set the SRQ line true upon receipt of the trailing edge of the multiprogrammer flag. As stated previously a multiprogrammer flag is only obtained when a "T" gate code is received by the 59500A. The three possible conditions which determine when the trailing edge of the flag will be received (with TME on) are as follows:

1. When using certain multiprogrammer input or output cards in serial (sequential) timing mode, TME must be programmed on (control word with a gate) before addressing a card with a gate. When the control word is received, the multiprogrammer will generate a handshake flag which will cause the 59500A to set the SRQ bus line true. Since this service request is caused by the control word, it must be cleared (see Paragraph 3-33) before addressing the card with a gate. Once the card has been addressed with a gate, it will control the flag line to the 59500A. When the 59500A receives the trailing edge of this flag, it sets SRQ true as an indication that the card has completed a data transfer.

2. When using multiple output cards in parallel (simultaneously) TME mode, the trailing edge of the multiprogrammer flag will not be returned to the 59500A until the last output card completes its data transfer operation.

3. When the multiprogrammer is in the interrupt mode (TME, $\overline{IEN}$ on), the multiprogrammer flag will not be received until a multiprogrammer card generates an interrupt. Note that for this condition (TME, $\overline{IEN}$ on), the NRFD line is not a function of the multiprogrammer flag. The TME and $\overline{IEN}$ control signals are sent to the timing logic (listen) to prevent the multiprogrammer flag from controlling the NRFD line in the interrupt mode. Thus, when cards interrupt, the resulting flag(s) will not affect the NRFD line, however, the trailing edge of the flag from the first interrupting card sets the SRQ bus line true.

3-30 Serial Poll

3-31 Serial poll is a method used by the calculator to determine which bus device has requested service (set SRQ bus line true). Essentially it consists of interrogating bus devices in sequence and reading back a status byte from each device which is used to identify the device(s) requesting service. When addressed to talk in the serial poll mode, the 59500A will return a status byte equal to decimal 64 if it is requesting service or a status byte of zero if it is not.

3-32 When a serial poll enable (SPE) command is received, it is latched in (see Paragraph 3-12) and sent to the encoder and the SRQ logic circuits. The SRQ logic circuit will send a status bit which indicates the state of the service request to the encoder. A logic 1 status bit, indicating SRQ is true, is translated into a decimal 64 by the encoder. A logic Ø status bit, indicating SRQ is false, is translated into a decimal Ø by the encoder.

3-33 When the serial poll enable latch is set and the 59500A talk address is received, the 59500A is placed in the serial poll active state (SPAS). For this condition, the SRQ logic circuit is reset and further transmission of SRQ on the HP-IB is inhibited. In addition, the 59500A transmits the status byte (decimal 64 or Ø) on data bus input/output lines DIO7-DIO1.

3-34 When the 59500A receives a serial poll disable command (SPD), an untalk command, or another devices talk address, the serial poll active state is terminated, preventing further transmission of the status byte.

3-35 3-Wire Handshake Process

3-36 As stated previously, a 3-wire handshake cycle occurs with each character transferred over the HP-IB data lines. The following paragraphs describe the 3-wire handshake process in greater detail. Figure 3-3 illustrates the 59500A interface with the DAV, NDAC, and NRFD HP-IB handshake lines. The handshake signal flow through the multiprogrammer system is also shown. A 3-wire handshake cycle occurs when the 59500A is receiving characters

(command and listen modes) and when the 59500A is sending data (talk mode).

3-37 Command Mode and Listen Mode Handshake Process. When the HP-IB is in the command mode (ATN line true) or when the 59500A is in the listen mode (addressed to listen and ATN line goes false), the listen handshake logic (see Figure 3-3) in the 59500A is enabled. As the 59500A receives each character (whether recognizable or not), a 3-wire handshake cycle with the HP-IB occurs.

3-38 The cycle is initiated when DAV goes true. The NDAC signal is controlled directly by the DAV signal through the listen handshake logic circuit. However, control of the NRFD signal depends upon the following conditions (refer to Figure 3-3):

1. If any character except a "T", "X", or "Z" is received, NRFD is controlled directly by the DAV signal. When the DAV signal goes true, a 3-wire handshake cycle occurs automatically between the talker (assume calculator) and the 59500A.

2. If a "T", "X", or "Z" gate code character is decoded, a delay circuit in the 59500A holds NRFD true for 30μsec. This delay prevents the calculator from inputting another character for 30μsec allowing the multiprogrammer sufficient time to process and store the existing data. As shown in Figure 3-3, a "T" code also generates the multiprogrammer gate which (when timing mode, TME is off) results in an automatic flag, from the 6940B handshake logic. However, since the 6940B handshake circuit provides a 30μsec (nominal) delay in parallel with the 59500A delay, the NRFD signal will still be held true for 30μsec (nominal).

3. If the multiprogrammer is in the timing mode (TME on) and a "T" is received, the 3-wire handshake cycle is not automatic but is dependent upon the flag signal from the 6940B multiprogrammer. In the timing mode, the multiprogrammer gate ($\overline{GTE}$), generated by the "T", is extended (DST, $\overline{GAT}$, signals) through the mainframe to the input/output card(s) and any associated external device(s) as shown in Figure 3-3. Timing circuit(s) in the input/output card(s) or external device(s) will, upon receipt of the gate signal, generate a flag ($\overline{FLG}$, $\overline{CTF}$) which will hold the NRFD bus line true until the input or output operations are completed. The flag line will override the 30μsec delay in the 59500A if the operation(s) are not completed within 30μsec. In many cases, the operation(s) require a number of seconds before completion. When the operation(s) are completed, the trailing edge of the flag will cause NRFD to go false. In effect by controlling the NRFD line with multiprogrammer flag, the multiprogrammer's 2-wire handshake (gate/flag) is converted to the HP-IB 3-wire handshake process (DAV, NDAC, NRFD). The only time the multiprogrammer flag will not affect the NRFD line is when the multiprogrammer is in the

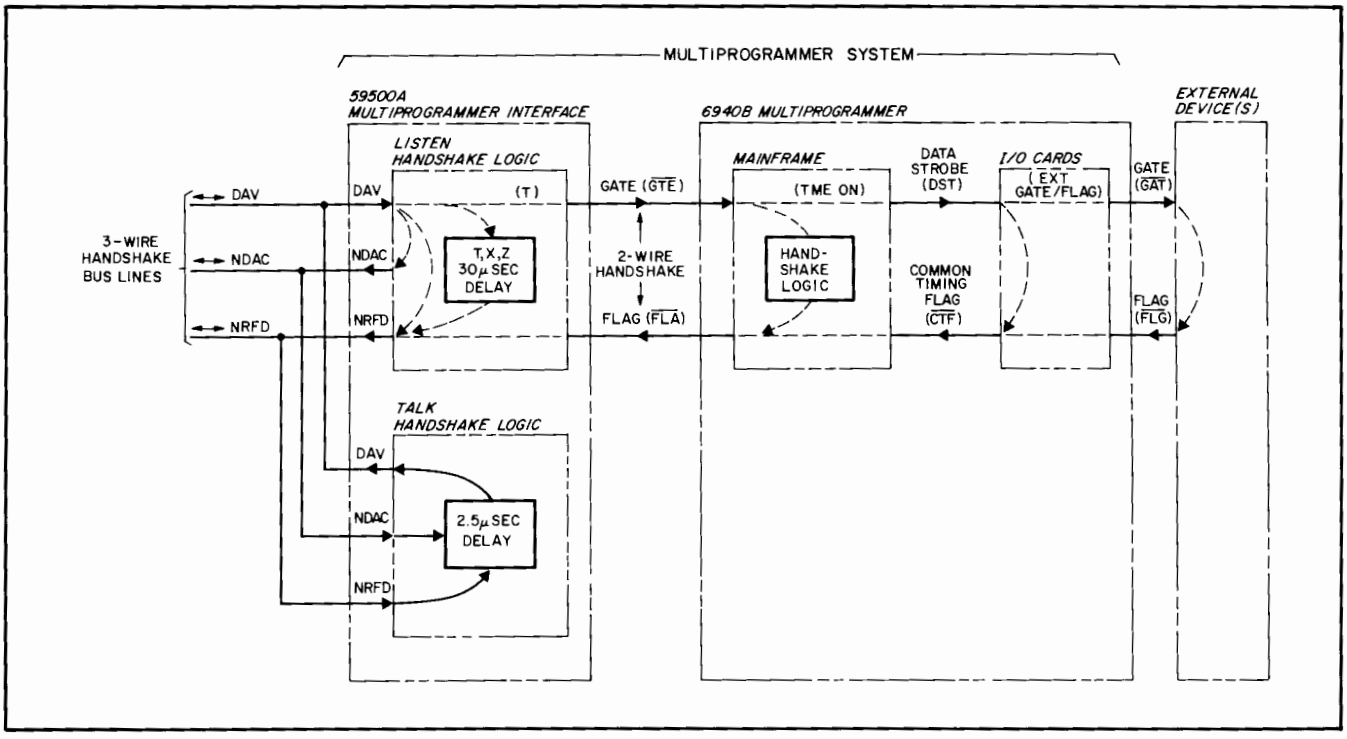


Figure 3-3. Multiprogrammer System Handshake Signal Paths

interrupt mode (TME, IEN both on). For this condition, NRFD is not a function of the flag line. The multiprogrammer's automatic and extended handshake processes are described in detail in Chapter II.

3-39 A 3-wire handshake cycle between the talker (calculator) and listener (59500A) occurs in the following sequence (refer to Figure 3-4A and 3-4B):

T1: Initially, the 59500A is ready to accept data; NRFD signal is false and the NDAC signal is true.

T2: Calculator puts a character on the HP-IB and indicates that the character is valid by setting DAV true.

T3: Upon sensing DAV true, the handshake logic (listen) circuit in the 59500A automatically sets NDAC false and NRFD true.

T4: Calculator senses NDAC false and sets DAV false indicating the character is no longer valid.

T5: Upon sensing DAV false, the handshake (listen) circuit in the 59500A automatically sets NDAC true and NRFD false (if character received is not a "T", "X", or "Z"). If a "T" (TME off), "X", or "Z" is received, the NRFD is held true for 30μsec. If a "T" (TME on) is received, NRFD is held true for 30μsec or more depending upon the time required to complete the input or output operation(s).

T6: With NRFD false, the calculator sets DAV true and another character is transferred (T_3 through T_5 are repeated).

3-40 Talk Mode Handshake Process. When the 59500A is in the talk mode (addressed to talk and the ATN line goes false), the talk handshake logic is enabled. As each character is transferred from the 59500A to the calculator, a 3-wire handshake cycle occurs. The sequence of operations is the same as described in Paragraph 3-39 and Figure 3-4A except in this case, the 59500A is the talker and the calculator is the listener. As shown in Figure 3-3, setting DAV true is delayed 2.5μsec to allow for HP-IB delays.

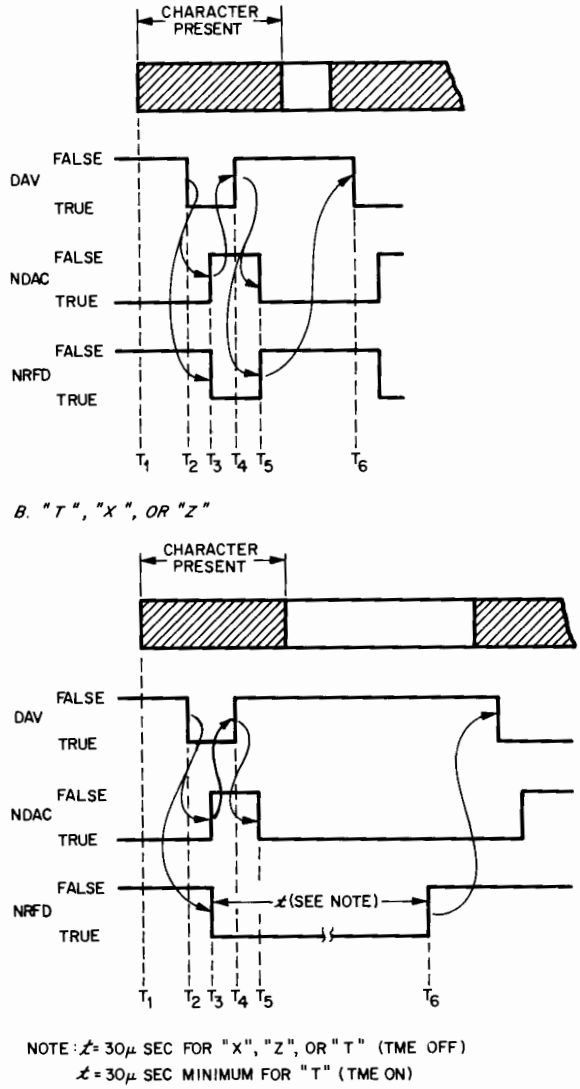


Figure 3-4. 3-Wire Handshake Sequence

Chapter IV
INSTALLATION

4-1 This chapter provides procedures for interconnecting a calculator and a multiprogrammer system using the HP-IB. The procedures include gathering the proper equipment, setting the appropriate address switches, connecting the cables, and turning on the equipment in the proper sequence.

4-2 EQUIPMENT REQUIRED

4-3 The following equipment is required to assemble the system:

1. Programmable Calculator HP 9820A, 9821A, or 9830A.

2. Bus Interface Kit HP 59405A: Includes the bus interface I/O card and the appropriate plug-in ROM. Peripheral Control II ROM 11224A is provided for HP 9820A/9821A. Extended I/O ROM 11272B is provided for HP 9830A. Note that the 11272B ROM is built into a 9830A equipped with Option 272.

3. Math ROM 11221A for 9820A/9821A: Required for the 9820A or 9821A calculators to properly program the multiprogrammer. The functions of the Math ROM are included within the 9830A.

4. Multiprogrammer Interface Unit HP 59500A.

5. Multiprogrammer HP 6940B (plus 6941B Extender Units if needed).

6. Input/Output Plug-In Cards for 6940B/6941B Multiprogrammer.

7. Interconnecting Cables:
   a. Calculator-to-59500A Interface Unit: Standard 72-inch (1.8 meters) HP-IB cable No. 10631B, supplied with 59500A.

   b. 59500A-to-6940B: Standard 18-inch (0.46 meters) chaining cable No. 14541A, supplied with 59500A.

   c. 6940B-to-6941B: Standard 18-inch (0.46 meters) chaining cable No. 14541A, purchased separately. Lengths up to 100-feet (30 meters) are available on special order.

4-4 INSTALLING ROM's AND BUS INTERFACE CARD

4-5 Install the ROM's (as required) in the calculator (see Figure 2-2 in the applicable HP-IB Calculator User's Guide). Next, install the bus interface card in any of the I/O slots on the rear of the calculator as shown in Figure 2-3 of the applicable HP-IB Calculator User's Guide.

4-6 SETTING ADDRESSES

4-7 The talk and listen addresses for the multiprogrammer system are selected by address switches on the rear of the 59500A interface unit. The switches are set to the suggested talk address of "W" and listen address of "7" as the unit is shipped from the factory. As shown in Figure 4-1, there are seven address switches. The last two switches, 6 and 7, are ignored. Check that switches 1 through 3 and 5 are set to "1" and switch 4 is set to "0" to select a talk address of "W" and listen address of "7". These multiprogrammer addresses are used in the sample programs and verification routines given in this User's Guide.

4-8 The bus interface card for the HP 9820A, 9821A, or 9830A calculator is shipped from the factory with a preset talk address of "U" and a preset listen address of "5". These calculator addresses are used in all sample programs and verification routines used in this guide.

NOTE

Before programming, write down all talk and listen addresses and the slot addresses of all cards used in the 6940B/6941B multiprogrammer. This will be helpful in programming and debugging.

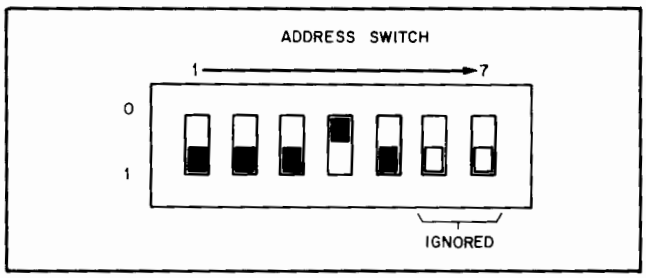


Figure 4-1. Address Switches on Rear of 59500A

4-9 CONNECTING CABLES

4-10 Connect cables as shown in Figure 4-2. Note that the 10631B HP-IB cable, supplied with the 59500A, is 6 feet in length. HP-IB cables are also available in 3 feet and 12 feet lengths. Refer to Pages 2-8 and 2-9 of the applicable HP-IB Calculator User's Guide for further information regarding HP-IB cables and length restrictions.

4-11 SYSTEM TURN-ON SEQUENCE

4-12 To prevent the random setting of the 59500A's service request (SRQ) and the multiprogrammer's system enable (SYE) signal at turn-on, the instruments must be turned-on in the following sequence:

1. Turn on calculator (and printer if you have one).
2. Set 6940B to LOCAL and then turn power on.
3. Turn on 59500A.
4. Now switch 6940B to REMOTE.

4-13 If desired, verify that the 59500A and 6940B (excluding I/O cards in 6940B) are properly connected by performing the verification procedures in Chapter V. Checkout procedures for each I/O card are provided in Chapter VI.

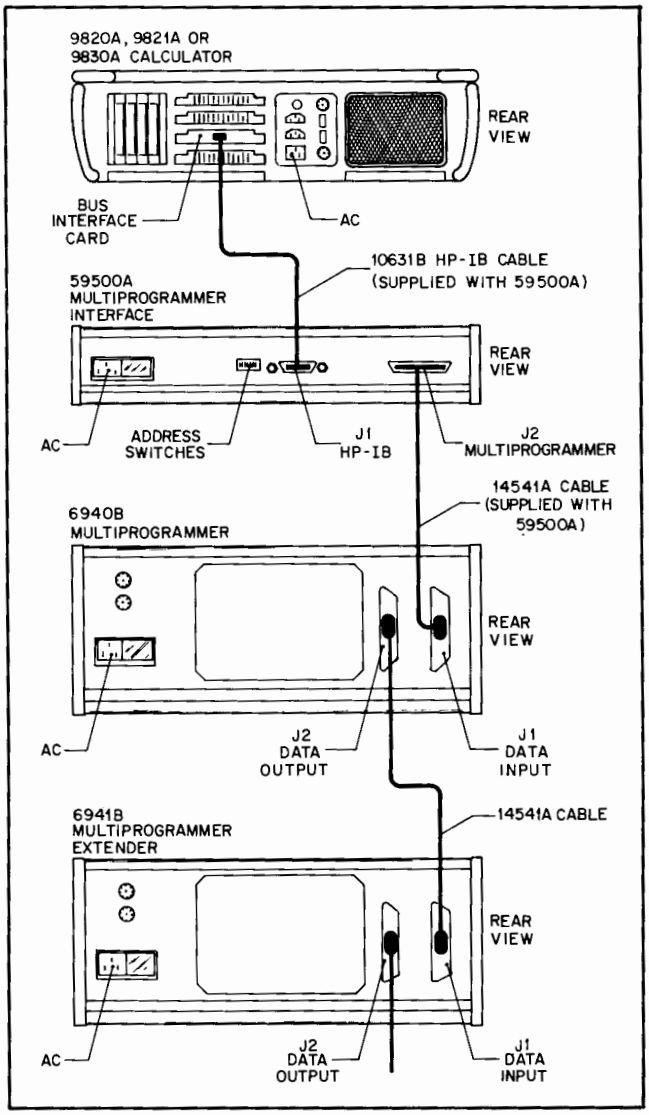


Figure 4-2. Interconnecting Cables

Chapter V
PROGRAMMING FUNDAMENTALS

5-1 This chapter outlines the fundamentals of programming the multiprogrammer system with the 9820/21A and 9830A calculators. The information presented in this chapter applies equally to all three calculators with specific differences being pointed out, where applicable. Included in this chapter are multiprogrammer compatibility with the HP-IB, how to program the basic word formats and verification programs for the 6940B multiprogrammer mainframe and 59500A interface unit.

5-2 HP-IB INTERFACE FUNCTIONS

5-3 The multiprogrammer system (Models 59500A and 6940B/6941B) is capable of performing the following functions on the HP-IB:

a. Talk
b. Listen
c. Service Request and Serial Poll

Conversely, the multiprogrammer system cannot act as a controller, extended talker, or extended listener and will not respond to remote-local, parallel poll, device clear, or device trigger commands.

5-4 MULTIPROGRAMMER WORD FORMATS

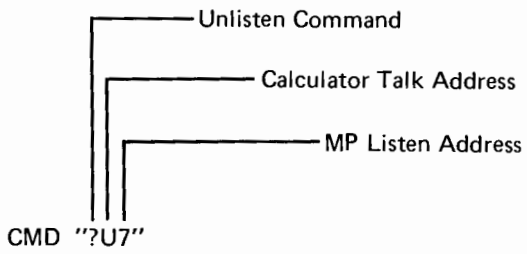
5-5 The next portion of this chapter is devoted to the fundamentals of communicating between the calculator and the multiprogrammer system (59500A/6940B, 6941B). For calculator-to-multiprogrammer exchanges, three different types of codes (words) can be sent and for multiprogrammer-to-calculator exchanges two types of words can be received. The following paragraphs describe the basic construction and purpose of each word type and how to send the output words or read in the input words. Each word type is treated separately without too much regard for the programming sequences which can be used to accomplish a specific task. Actual programs, using these basic word formats, are included in the next chapter.

5-6 Calculator Output Words

5-7 Three types of output words are sent from the calculator to the multiprogrammer system: control words, data words, and address words.

5-8 Multiprogrammer Listen Address. Before any of these words are sent, a command statement must be issued to establish the calculator as the talker and the multiprogrammer as the listener. Throughout this chapter it is assumed that the calculator has been assigned its standard listen and talk addresses of "5" and "U", respectively, and that the multiprogrammer system has been assigned a listen address of "7" and talk address of "W". Using these addresses, the addressing sequence would be as follows:

Example 1: Addressing the Multiprogrammer to Listen:



5-9 Control Word. Usually, the first word sent to a multiprogrammer system will be a control word. It is used to specify a unit address and the operating mode of the multiprogrammer system. The unit address portion selects which mainframe in the chain (of one 6940B Multiprogrammer and up to fifteen 6941B Extenders) will respond to subsequent data or address words from the calculator. The mode portion of the word selects one of several output or input modes of multiprogrammer operation (System Enable, Timing Mode Enable, Input Select, etc.). How the multiprogrammer system responds to these various control word modes is described in Chapter II of this guide, as well as in the 6940B Service Manual. A good understanding of these modes is essential for effective programming involving specific applications.

5-10 Figure 5-1 illustrates the structure of a control word as transmitted on the HP-IB. The six control word characters are transmitted and obeyed in the order written, from left-to-right. The address character "0", is the control word tag that tells the 6940B unit that the data characters that follow are to be interpreted as control mode and unit address information.

5-11 The data field consists of four octal numbers. The most significant digit, D_4, conveys no information in a control word and can be sent as a "Ø" or not transmitted at all (see note on following page).

NOTE

Because of the way the multiprogrammer system processes the octal characters in a data field, any leading "Ø" 's need not be sent. Of course, trailing "Ø" 's must be transmitted.

Digits D_3 and D_2 specify the operating mode as indicated by the codes shown in Table 5-1. Only the most commonly used codes are shown in Table 5-1; but combinations for other codes are possible and become obvious through inspection.

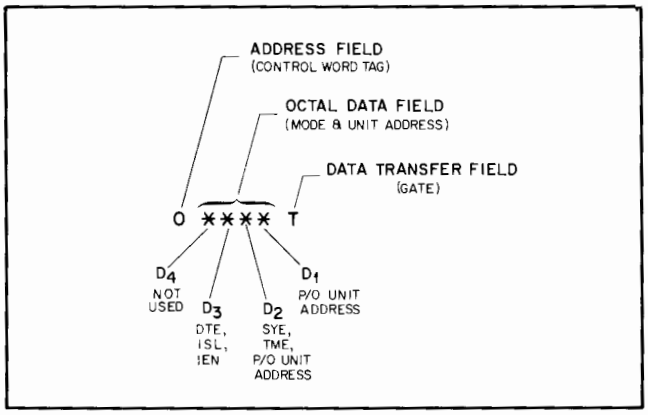


Figure 5-1. Control Word

Table 5-1. Operating Mode Codes

Multiprogrammer Operating Mode		Data Field*
Output Mode	**Input Mode**	
SYE Off		ØØØØ
SYE On		ØØ4Ø
DTE, SYE On		Ø14Ø
DTE, SYE, TME On		Ø16Ø
	ISL, SYE On	Ø24Ø
	ISL, SYE, TME On	Ø26Ø
	IEN, SYE, TME On	Ø46Ø

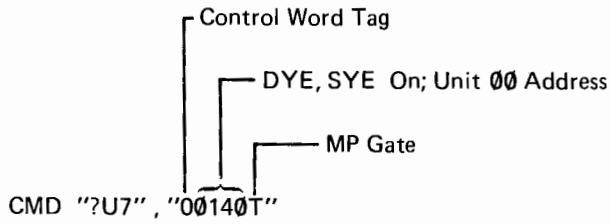
*Data field codes shown are valid for a unit address of ØØ only.

5-12 Digits D_2 and D_1 specify the unit address. The first unit in the chain (6940B, Master) has an address of ØØ, the second unit (6941B Extender) has an address of Ø1, and so forth, up to a unit address of 15. The two unit address digits are simply added (in octal) to the last two control mode digits shown in Table 5-1 to obtain the final data field code. For example, if the user wants to select unit number one (second unit in chain) and program the DTE, SYE, and TME modes, he would send a code of Ø161 in the data field. To select unit number 15 (last unit) and the same modes, he must send an Ø177 ($Ø16Ø_8 + 17_8$).

5-13 The last character in a control word must be a "T". The multiprogrammer system interprets this character as a "gate" which initiates the gate/flag handshake sequence within the 6940B multiprogrammer.

5-14 A typical control word, with DTE and SYE on, would be sent as follows:

Example 2: Typical Control Word



5-15 Data Word. Data words select and control output cards and a few "special purpose" input cards. The data word selects an individual card within the mainframe previously specified by a control word.

5-16 Figure 5-2 shows the basic construction of a data word sent from the calculator. The address character selects the particular card (one of 15 slots in a mainframe) that will receive the four "control" digits in the data field portion of the word. Table 5-2 shows the card slot address that corresponds to each ASCII address character. The data field consists of four octal digits that can be of any value between ØØØØ and 7777 (Appendix A contains an explanation and subroutines on how to convert numbers from Octal to decimal and vice-versa). As in a control word, leading Ø's need not be sent in the data field. A data word with ØØØØ in the data field could be sent with just an address and a gate (T) character; e.g., to send all Ø's to slot 401, simply transmit "AT". Of course, this word could also be sent as "AØØØØT".

5-17 The data transfer character must always be a "T" (gate) in a data word, the same as in a control word.

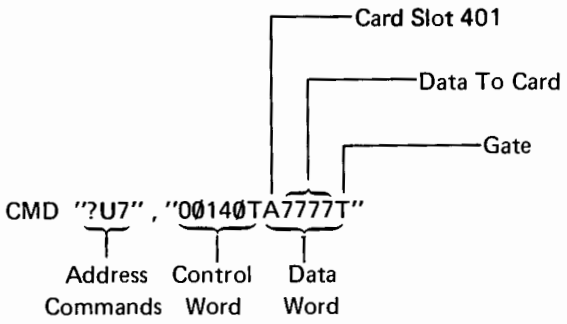
5-18 Example 3 shows a command statement using a typical data word preceded by the sample control word shown previously.

Table 5-2. Card Slot Addresses

ASCII ADDRESS CHARACTER	ASCII CODE	MULTIPROGRAMMER CARD SLOT ADDRESS
@ *	1ØØØØØØ	Slot 400
A	1ØØØØØ1	Slot 401
B	1ØØØØ1Ø	Slot 402
C	1ØØØØ11	Slot 403
D	1ØØØ1ØØ	Slot 404
E	1ØØØ1Ø1	Slot 405
F	1ØØØ11Ø	Slot 406
G	1ØØØ111	Slot 407
H	1ØØ1ØØØ	Slot 408
I	1ØØ1ØØ1	Slot 409
J	1ØØ1Ø1Ø	Slot 410
K	1ØØ1Ø11	Slot 411
L	1ØØ11ØØ	Slot 412
M	1ØØ11Ø1	Slot 413
N	1ØØ111Ø	Slot 414

* To send an @ character, press the SHIFT and RESULT keys (9830A Calculator) or the GO TO key (9820/21A Calculators).

Example 3: Typical Data Word:



In this example, an output card (in slot 401, Unit ØØ) is being directed to output data represented by the octal number "7777".

5-19 Address Word. An address word selects the input card that is to send data (a return data word) back to the calculator. Before an address word can be sent, a control word must have first selected an input operating mode (and the unit address). Notice also that before the addressed input card can send a return data word back to the calculator, the multiprogrammer system must be placed in the talk mode (refer to Paragraph 5-23; calculator input words).

5-20 An address word contains just two characters (Figure 5-3). The address character specifies one of 15 possible input card slot addresses which are identical to those given in Table 5-2 for a data word. The data transfer field character can be a "T" (read with gate) an "X" (read without gate), or a "Z" (read live data). Which character to use depends on the specific application and the input card type (refer to Chapters II and VI). A "T" always results in the generation of a "gate" to the addressed input card, and the data from the input card (return data word) will be stored in the 59500A on the trailing edge of the flag from the multiprogrammer. Thus, a "T" character is used when it is desired to wait for the completion of a gate/flag sequence before storing the return data word in the 59500A. As previously mentioned, to obtain the return data information

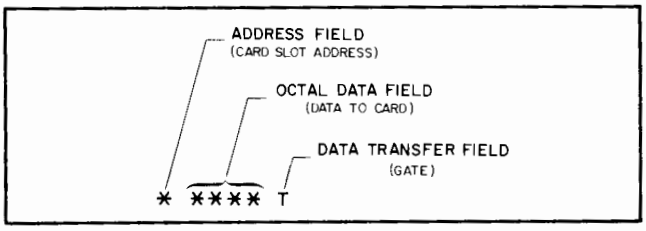


Figure 5-2. Data Word

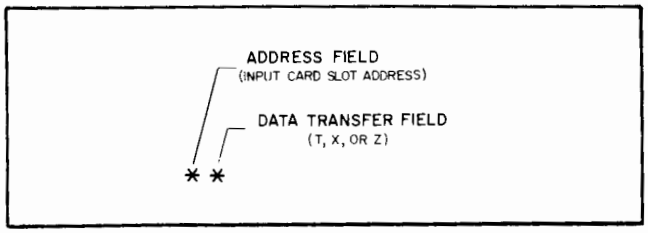


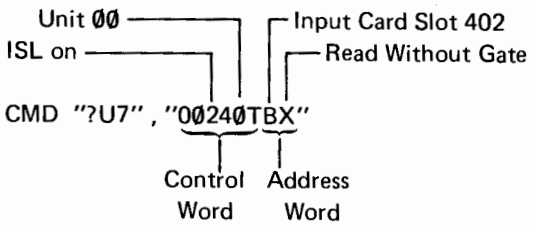
Figure 5-3. Address Word

after an address word is sent, the user must first address the multiprogrammer to talk and then follow with an enter (read) statement.

5-21 An "X" character is sent when it is necessary to store the return data word from the addressed card immediately, without waiting for a multiprogrammer gate/flag cycle. The "X" character itself, stores the return data in the 59500A as soon as it is received from the input card. The "Z" character allows the return data from the addressed card to continuously pass through the 59500A to its' output encoder, without being stored. To obtain the "live" data, the user can simply turn the attention line on and off (send a CMD" ") and then read in the data. When using the "Z" character, the user must keep in mind that the return data may be continuously changing and can therefore be difficult to interpret.

5-22 Example 4 shows a typical address word sent to an input card installed in multiprogrammer slot 402 (address character B). Notice that the preceding control word calls for an input mode of operation (ISL on), and the address word "BX", specifies a read without gate sequence.

Example 4: Typical Address Word:



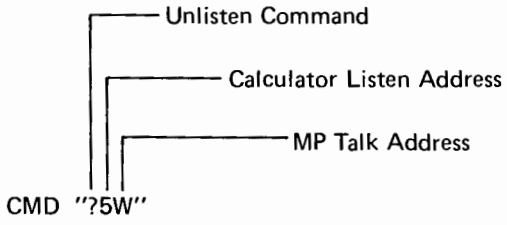
5-23 Calculator Input Words

5-24 Two types of input words are sent to the calculator from the multiprogrammer system: return data words and status characters (bytes).

5-25 **Multiprogrammer Talk Address.** Before the calculator can receive a return data word, a command statement must be sent directing the calculator to listen and the multiprogrammer to talk. Example 5 shows the necessary address commands.

5-26 **Return Data Word.** Return data words contain information from an input card previously selected by an address word. Each word (Figure 5-4) contains an input request character, four octal data characters, and a carriage return/line feed (not shown on Figure 5-4).

Example 5: Addressing the Multiprogrammer to Talk:



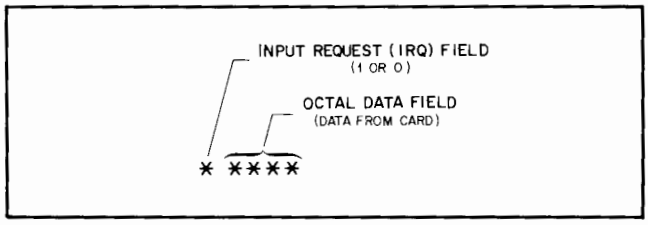


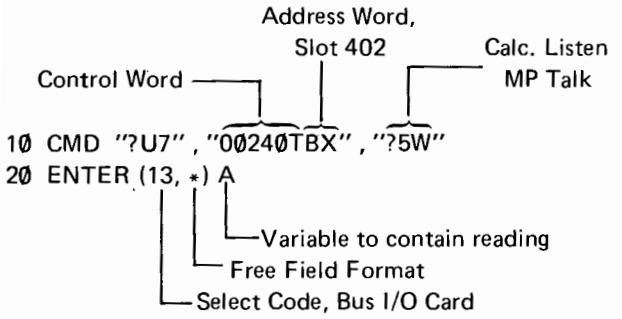
Figure 5-4. Return Data Word

5-27 The input request (IRQ) character can be either a "1" or a "Ø". A "1" indicates that the input card has returned a flag and its input data is valid. Moreover, during an interrupt sequence (TME and IEN on), a "1" in the IRQ field indicates that this card has generated an interrupt. A "Ø" indicates that: (1) the input card has not returned a flag (card is still busy or has not interrupted) and its' data is not valid; or (2) the card does not contain an IRQ function. Some input cards; such as the voltage monitor card (69421A), do not contain an IRQ function and a "Ø" in the IRQ field can be ignored.

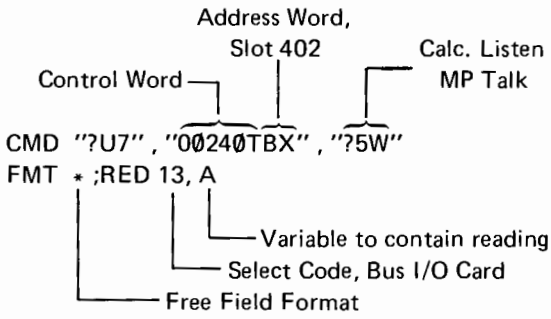
5-28 The four data characters, representing the input information from the multiprogrammer card, can be any octal number, between ØØØØ and 7777. Hence, a return data word with all "7" 's in the data field and a "1" in the IRQ field would appear as a 17777 on the calculator display. Example 6 shows how to enter (read in) a return data word and store it in the variable "A" register of the calculator. First, a CMD statement is sent containing a control word with ISL on. Next, an address word is sent to select the input card in slot 402 (Unit ØØ) to return its' data. After the necessary talk/listen commands ("5W"), an ENTER statement (9830A) or a READ statement (9820A/21A) is sent, allowing the return data to be stored in variable "A". The return data in variable "A" can be utilized directly unless it is obtained from a card that contains the IRQ function. In these cases, the return data must first be checked to determine if there is a "1" in the IRQ field. If the IRQ character is a "1", it must then be subtracted from the input data before continuing with the program. How to check for IRQ and subtract it, if necessary, is shown in the applicable input card programs of the next chapter.

Example 6: Addressing an Input Card and Entering its' Return Data:

9830 Calculator



9820/21 Calculators



5-29 **Status Bytes.** The 59500A of the multiprogrammer system returns an 8-bit status byte to the calculator in response to a serial poll routine. As explained in Chapter 5 of the calculator users guides, serial polling is the method used by the calculator to determine which bus device has requested service (set SRQ true). The multiprogrammer will request service (set SRQ line) only in the timing mode (TME on) upon completion of a multiprogrammer flag. Both input and output card types can set SRQ and Chapter III lists the specific operating conditions required for a service request to occur.

5-30 To check the bus for the presence of a service request the calculator uses a status check. A status check code of "0" or "1" indicates that one or more devices on the bus have requested service. A status code of "2" or "3" means that service was not requested. Thus, if a status check indicates a "0" or "1" condition (SRQ set), a serial poll can be done to identify the source of the service request. Appendix A contains a serial poll subroutine that may be used to poll the multiprogrammer system. Briefly, this subroutine places the multiprogrammer system in the serial poll mode, addresses it to talk, and then reads its' status byte into calculator variable A4 (9830) or R4 (9820/21A). The status byte stored in the variable will have a decimal value of 64, if the multiprogrammer requested service; or a value of 0, if it had not requested service. The subroutine of Appendix A also clears the SRQ line so that the calculator can detect the presence of new service requests.

5-31 Example 7 shows a sample program for a status check of the bus and a serial poll of the multiprogrammer.

5-32 MULTIPROGRAMMER SYSTEM VERIFICATION

5-33 The following portion provides procedures to verify that the major functions of the 59500/6940 system can be programmed from your HP calculator. Since there are significant differences in programming a 9830 calculator versus the 9820/9821 calculators, the detailed program listings are repeated for each of these calculator types. However, since the same verification checks are made regardless of calculator type, the general descriptions and operating procedures for the verification tests, apply to users with any of the three calculator types.

5-34 Description Of Tests

5-35 The verification programs establish that the major functions of the 59500/6940 multiprogrammer HP-IB system are operational. The tests are essentially "GO/NO-GO" checks with hardware diagnoses of inoperable functions left to the Operating and Service manuals for the 59500 and 6940 equipment. It is recommended that the tests be executed in the order presented in the following paragraphs the first time the 59500/6940 system is placed in operation. Later, after the system has been operational, the user can select any of the test(s) to verify function(s) that are suspected of being faulty.

5-36 **59500 LISTEN Mode Verification.** This program verifies the following listen mode capabilities of the 59500:

1. The 59500 responds to its listen address (assumed to be "7").
2. The 59500 unlistens when appropriately programmed and does not respond to any other listen addresses.
3. The 59500 does not unlisten when any other listen address is programmed.
4. The 59500 unlistens in response to interface clear.
5. The 59500 unlistens when made a talker.

5-37 **59500 TALK Mode Verification.** This program verifies the following talk mode capabilities of the 59500:

1. The 59500 responds to its talk address (assumed to be "W").
2. The 59500 untalks when programmed to untalk.
3. The 59500 untalks when any other talk address programmed.

Example 7: Bus Status Check and Multiprogrammer Serial Poll:

9830A Calculator

```
110 IF STAT 13>1 THEN 150
120 GOSUB 2000
130 IF A4#64 150
140 REM SRQ SET-TAKE APPROPRIATE ACTION
150 REM CONTINUE PROGRAM
```

9820/21A Calculators

```
IF RDS 13>1; GTO "CONT"
GSB "SPOLL"
IF R4≠64; GTO "CONT"
   (SRQ SET-TAKE APPROPRIATE ACTION)
"CONT" CONTINUE PROGRAM
```

Explanation:

110: Bus Status Check, if service not requested, go to continue main program, line 150.
120: Service requested; go to serial poll subroutine for multiprogrammer.
130: Examine MP status byte in register; if not equal to 64, service not requested, go to continue program, line 150.
140: Multiprogrammer service request. Take appropriate action; e. g., read input card data, start "interrupt search" of input cards, send new data to output card, or ignore service request.

4. The 59500 untalks when made a listener.
5. The 59500 untalks in response to interface clear.

5-38 **59500/6940 Serial Poll Mode Verification.** This program verifies the following serial poll mode capabilities of the 59500/6940:

1. The 6940/59500 correctly set the service request line.
2. The 59500 executes a serial poll correctly, resetting its service request appropriately.
3. The 59500 disables the serial poll mode when appropriately programmed.
4. The 59500 returns the correct status byte when serial polled. Checks are made that status bit 7 of the status byte is on if service request is set and off if service request is cleared prior to the serial poll.
5. The 59500 does not reset its service request in response to interface clear.

5-39 **Data Output – Calculator to 59500/6940 – Verification.** This program verifies the following data output capabilities of the 59500/6940:

1. When programmed to listen, the 59500 correctly outputs valid address/data to the 6940. Valid 59500/6940 address/data patterns are manipulated to verify that the 16 data bits output from the 59500 to the 6940 can be correctly programmed.
2. The program verifies that when the 59500 is not a listener, it does not process valid address/data for output to the 6940.
3. The program also verifies that when it is a listener, the 59500 correctly handshakes but does not process output bus data that is not valid for the 59500/6940. That is, the 59500 does not process or output to the 6940 bus data that is not: (1) a valid 6940 address byte; (2) an octal data byte; or (3) an X, T, or Z data transfer code.

5-40 **Data Input – 59500/6940 to Calculator – Verification.** This program verifies the following data input capabilities of the 59500/6940:

1. The program instructs the user on how to input data to the calculator from the 6940 via the 59500, sets the 59500/6940 to talk, and then verifies that the input is correct. The user can input as many data patterns as desired.
2. The program also verifies that the 59500 does not accept data from the 6940 when the 6940 Flag input is not set with the input data.

5-41 **59500/6940 Gate/Flag Verification.** This program verifies the following Gate/Flag features of the 59500/6940:

1. The 59500 correctly sets its Gate output to the 6940 and the 6940 correctly receives the Gate when a "T" is programmed.
2. The 59500 resets the Gate to the 6940 when an "X" is programmed.
3. The 6940 Flag to the 59500 is correctly received in the 59500.

5-42 System Installation Requirements

5-43 The Calculator – 59500/6940 System is connected as described in Chapter IV of this guide. This chapter also describes the calculator options that are required to program the 59500/6940 system. In addition, the verification programs all assume that the 59500/6940 has been assigned a listen address of "7" and talk address of "W". Note, similarly, that the verification programs also assume that the calculator has been assigned its standard listen address of "5" and talk address of "U". Finally, these verification programs are designed to exercise the functions of the 59500/6940 only; do not install any multiprogrammer input/output cards in the 6940 mainframe. Further, do not connect

any 6941 extenders to the 6940 during the system verification.

5-44 The 9830 – 59500/6940 verification programs assume that you have a printer in your calculator installation; if you don't have a printer, you can modify the programs by changing all PRINT statements to DISP. (see Paragraph 5-49).

5-45 What To Do In Case Of Trouble

5-46 All of the verification programs are user-interactive, functional tests of the major functions of the 59500/6940. The programs continually suspend operation (stop) and request (either via the 9830 display or the 9820/21 printer) that the user examine specific test status (usually indicators on the 59500/6940) and continue the verification program if they are normal. If test results are not as requested by the program, the user should clear up the problem and re-run the test before continuing with subsequent verification programs. While the failure of a test most often indicates that a hardware problem exists, certain operational or equipment configuration mistakes can be made and result in erroneous indications. If you get an indication of a failure, run through the following checklist before initiating the hardware troubleshooting procedures given in the Operating and Service Manuals.

1. Check that the required calculator options have been installed as described in Chapter IV.

2. Check that the interconnecting cables between the calculator, 59500, and 6940 are installed correctly.

3. Check that all units are turned on and that the 6940 is set to REMOTE operation unless otherwise instructed in a verification program (all programs except the Data Input and Gate/Flag verification programs require that the 6940 be set to REMOTE operation only).

4. Check that all multiprogrammer input/output cards have been removed from the 6940 and that there is no 6941 connected to the 6940.

CAUTION

Turn power off at the 6940 before removing I/O cards. Failure to do so may result in equipment malfunctions.

5. Check that there are no other devices connected on the HP-IB.

6. Check that the 59500 address switches have been set for listen address "7" and that the calculator bus interface has been set for listen address "5". If either or both of these address assignments must be changed, then you must modify the verification programs accordingly.

7. Check that the program was typed into the calculator correctly. For instance, on the 9830 if you type "U" with the SHIFT key depressed, you will enter a lower-case "u" instead of the talk address of the calculator. The calculator display (and printer, if you have one) does not print lower-case letters, however, so that you cannot detect this error through inspection . If in doubt, re-enter the program(s).

8. Check that the 59500 and 6940 were turned on in the sequence given in the verification program. The turn on sequence is especially important for the Serial Poll Mode, Data Output, and Data Input tests.

5-47 Operating Procedures – 9830 Calculator – 59500/6940 Verification.

5-48 The verification program listings for a 9830 calculator-based HP-IB system employing the 59500/6940 are included immediately following Paragraph 5-51. To execute the programs, simply type them in as listed and depress the RUN, EXECUTE calculator keys. Note that the programs do not overlay themselves so that it is possible to load all of them together.

5-49 Once the verification programs are loaded into the calculator, they are essentially "stand alone" in that all instructions needed to execute the program are given on the calculator Printer and Display. As previously mentioned, if you don't have a printer, you must modify the 9830 verification programs. One alternative is to change all PRINT statements to DISP in order to view the messages on the calculator display (putting in appropriate STOPs, of course, for multi-line messages). The other alternative is to simply delete all PRINT statements and refer to the program listings for program instructions. As part of program execution, the user notifies the program that correct test results have been achieved by pressing calculator keys, CONT, EXECUTE and continuing the program. Thus, there are two types of messages, described below, displayed for the user by the program.

5-50 **Program Instruction Messages.** These messages are used by the program to describe the test results the user should observe or how to implement a test procedure. All of these messages are defined by the addition of a "(C)" at the end of the line. The "(C)" indicates that the user should continue the program by depressing calculator keys CONT, EXECUTE.

5-51 **Test Inquiry Messages.** These messages are all single-line display messages that ask the user to observe specific test results as indicated by the 59500/6940 indicators. These messages are all yes/no questions that the user must respond to in either of two ways:

1. Yes. If the test results are as indicated in the message, the user can press CONT, EXECUTE to so notify the program and continue the test. Note that only the indicators specifically mentioned in the message are significant so that the state of any other 59500/6940 indicators

should be ignored.

2. No. If the expected test results are not obtained, the user should not continue but should fix the problem (see operational checklist). If you are suspicious of the results, you can try to rerun the test that failed from the beginning to reverify that you have a problem.

9830A VERIFICATION PROGRAMS

```
1 PRINT "59500 LISTEN MODE VERIFICATION. FOLLOW DISPLAY INSTRUCTIONS."
3 PRINT
5 DISP "TURN 59500 OFF,THEN BACK ON.(C)"
7 STOP
9 CMD "?U7"
11 DISP "59500 LISTEN LITE ON?"
13 STOP
15 CMD "?!#$%&'()*+,-./0123456789:;=<>"
17 OUTPUT (13,19)256,34,512;
19 FORMAT 3B
21 DISP " LISTEN LITE OFF?"
23 STOP
25 CMD "7!#$%&'()*+,-./12345689:;<=>"
27 OUTPUT (13,19)256,34,512;
29 DISP "LISTEN LITE ON?"
31 STOP
33 DISP "DEPRESS 'STOP,CONT,EXECUTE'"
35 STOP
37 DISP "LISTEN LITE OFF?"
39 STOP
41 CMD "7W"
43 DISP "LISTEN LITE OFF,TALK LITE ON?"
45 STOP
47 CMD "7"
49 DISP "LISTEN LITE ON,TALK LITE OFF?"
51 STOP
53 PRINT "LISTEN MODE O.K.-CHECK TALK MODE"
54 PRINT
55 END
```

```
100 PRINT "TALK MODE VERIFICATION. FOLLOW DISPLAY INSTRUCTIONS."
102 PRINT
104 DISP "TURN 59500 OFF,THEN BACK ON.(C)"
106 STOP
108 OUTPUT (13,110)256,95,512;
110 FORMAT 3B
112 CMD "W"
114 DISP "59500 TALK LITE ON?"
116 STOP
118 OUTPUT (13,110)256,95,512;
120 DISP "TALK OFF?"
122 STOP
124 DISP "TALK LITE SHOULD BLINK ONCE(C)"
126 STOP
128 Q=64
130 FOR N=1 TO 30
```

Talk Mode Verification Continued

```
132 IF Q=87 THEN 142
134 CMD "W"
136 OUTPUT (13,110)256,0,512;
138 DISP "TALK BLINK ON/OFF ONCE?"
140 STOP
142 Q=Q+1
144 NEXT N
146 CMD "W?"
148 DISP "TALK LITE OFF,LISTEN LITE ON?"
150 STOP
152 CMD "W"
154 DISP "DEPRESS 'STOP,CONT,EXECUTE'"
156 STOP
158 DISP "TALK LITE OFF?"
160 STOP
162 PRINT "TALK MODE O.K.-CHECK SERIAL POLL MODE."
163 PRINT
164 END
```

```
200 PRINT "SERIAL POLL MODE VERIFICATION. FOLLOW DISPLAY INSTRUCTIONS."
202 PRINT
204 DISP "TURN 59500 OFF,THEN BACK ON.(C)"
206 STOP
208 CMD "?U7","00020T"
210 DISP "59500 SERVICE REQUEST ON?"
212 STOP
214 IF STAT13 <= 1 THEN 224
216 DISP "59500 SRQ MALFUNCTION"
218 GOTO 270
220 FORMAT 5B
222 FORMAT 3B
224 OUTPUT (13,220)256,95,53,24,512;
226 CMD "W"
228 DISP "TLK & SER POLL ON,SERV.REQ. OFF?"
230 STOP
232 A=RBYTE13
234 IF A=64 THEN 240
236 DISP "STATUS BYTE ERROR-BIT 7 OFF"
238 GOTO 270
240 OUTPUT (13,222)256,25,512;
242 DISP "SERIAL POLL LITE OFF?"
244 STOP
246 OUTPUT (13,220)256,95,53,24,512;
248 CMD "W"
250 A=RBYTE13
252 IF A#64 THEN 258
254 DISP "STATUS BYTE ERROR-BIT 7 ON"
256 GOTO 270
258 CMD "?U7","00020T"
260 DISP "DEPRESS 'STOP,CONT,EXECUTE'"
262 STOP
264 DISP "SERVICE REQUEST LITE ON?"
266 STOP
268 PRINT "SERIAL POLL MODE O.K.-CHECK DATA OUTPUT."
269 PRINT
270 END
```

```
300 PRINT "DATA OUTPUT-9830 TO 6940-VERIFICATION.FOLLOW DISPLAY INSTRUCTIONS.
304 PRINT
306 PRINT "TURN 6940 OFF.TURN 59500 OFF,THEN BACK ON.TURN 6940 ON.(C)"
307 PRINT
308 STOP
310 CMD "?U7","O7777"
312 DISP "ALL 6940 DATA LITES ON?"
314 STOP
316 CMD "?U7","@"
318 DISP "ALL 6940 DATA LITES OFF?"
320 STOP
322 REM MARCH A '1' DOWN EACH 6940 BIT POSITION.
324 N=0
326 FOR A=0 TO 3
328 FOR B=0 TO 2
330 FORMAT F1005.0
332 Q=(2↑B)*(10↑A)
334 OUTPUT (13,330)"@"Q
336 DISP "ONLY 6940 DATA LITE"N"ON?"
338 STOP
340 N=N+1
342 NEXT B
344 NEXT A
346 CMD "?U7","A"
348 DISP "ONLY 6940 DATA LITE 12 ON?"
350 STOP
352 OUTPUT (13,*)"B"
354 DISP "ONLY 6940 DATA LITE 13 ON?"
356 STOP
358 OUTPUT (13,*)"D"
360 DISP "ONLY 6940 DATA LITE 14 ON?"
362 STOP
364 OUTPUT (13,*)"H"
366 DISP "ONLY 6940 DATA LITE 15 ON?"
368 STOP
370 REM MARCH A '0' DOWN BITS 15-00
372 X=(7777)-(2↑B*10↑A)
374 CMD "?U7","G7777"
376 DISP "ONLY 6940 DATA LITE 15 OFF?"
378 STOP
380 OUTPUT (13,*)"K7777"
382 DISP "ONLY 6940 DATA LITE 14 OFF?"
384 STOP
386 OUTPUT (13,*)"M7777"
388 DISP "ONLY 6940 DATA LITE 13 OFF?"
390 STOP
392 OUTPUT (13,*)"N7777"
394 DISP "ONLY 6940 DATA LITE 12 OFF?"
396 STOP
398 N=11
400 FOR A=3 TO 0 STEP -1
402 FOR B=2 TO 0 STEP -1
406 X=(7777)-(2↑B*10↑A)
408 OUTPUT (13,330)"O"X
410 DISP "ONLY 6940 DATA LITE"N"OFF?"
412 STOP
```

```
414 N=N-1
416 NEXT B
418 NEXT A
420 CMD "?U7","E2525"
422 DISP "ONLY EVEN(0,2,...)LITES ON?"
424 STOP
426 CMD "?U7","J5252"
428 DISP "ONLY ODD DATA LITES ON?"
430 STOP
432 REM VERIFY THAT 59500 DOES NOT SEND DATA
434 REM TO 6940 WHEN NOT A LISTENER.
436 CMD "?U7","@0000"
438 CMD "?U","O7777"
440 DISP "ALL 6940 DATA LITES OFF?"
442 STOP
444 REM VERIFY THAT WHEN A LISTENER,59500 H/S BUT DOES NOT PROCESS
446 REM NON-ADDRESS OR DATA CHARACTERS.
448 CMD "?U7","O7777"
450 FOR I=1 TO 99
452 FORMAT 1B
454 READ B
456 OUTPUT (13,452)B;
458 DATA 0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20
460 DATA 21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42
462 DATA 43,44,45,46,47,56,57,58,59,60,61,62,80,81,82,83,85,86
464 DATA 89,91,92,93,94,96,97,98,99,100,101,102,103,104,105,106,107
466 DATA 108,109,110,111,112,113,114,115,116,117,118,119,120,121,122,123
468 DATA 124,125,126,127
470 NEXT I
472 DISP "ALL 6940 DATA LITES ON?"
474 STOP
475 REM REPEAT ABOVE WITH 6940 DATA LITES OFF
476 OUTPUT (13,*)"@"
478 FOR I=1 TO 99
480 RESTORE
482 READ B
484 OUTPUT (13,452)B
486 NEXT I
488 DISP "ALL 6940 DATA LITES OFF?"
490 STOP
492 PRINT "DATA OUTPUT O.K.-CHECK DATA INPUT"
493 PRINT
494 END
```

```
500 PRINT "DATA INPUT-6940 TO 9830-VERIFICTION.FOLLOW PRINTER INSTRUCTIONS."
503 PRINT
504 PRINT "TURN 6940 OFF.TURN 59500 OFF,THEN BACK ON.TURN 6940 ON.(C)"
506 PRINT
508 STOP
510 OUTPUT (13,*)"OT@"
512 PRINT "SET 6940 TO LOCAL MODE. WHEN 'ENTER DATA' APPEARS,"
514 PRINT "INPUT DATA FROM 6940: 1.SET BITS 15,11-0 AS DESIRED,"
516 PRINT "2.PUSH 6940 'RETURN DATA'"
518 PRINT "3.DEPRESS 'CONT,EXECUTE' AND CHECK PRINTER.IF:"
520 PRINT "A.INPUT O.K. GO TO STEP 4. B.INPUT N.G. FIX PROBLEM."
522 PRINT "4(A)T0 INPUT DATA AGAIN,PRESS'CONT,EXECUTE' TO REPEAT FROM STEP 1.
524 PRINT " (B)T0 CONTINUE INPUT TEST, PRESS 'STOP, RUN 584, EXECUTE'.(C)"
525 PRINT
526 STOP
568 PRINT "ENTER DATA!"
569 PRINT
570 STOP
572 CMD "?W5"
576 ENTER (13,*)A
578 PRINT A
579 PRINT
580 STOP
582 GOTO 568
584 PRINT "SET 6940 DATA TO 7777 AND PUSH 'RETURN DATA,CONT,EXECUTE'"
585 PRINT
586 STOP
592 CMD "?W5"
594 ENTER (13,*)A
596 PRINT "WHEN'CHANGE DATA!' APPEARS:"
600 PRINT "1.SET BITS 11-0 AS DESIRED"
604 PRINT "2.DON'T PUSH 'RET DATA'"
608 PRINT "3.DEPRESS 'CONT,EXECUTE' AND CHECK PRINTER.IF:"
612 PRINT "A.INPUT DID NOT CHANGE,'CHANGE DATA!' APPEARS AGAIN"
616 PRINT "TO CHANGE DATA AGAIN, REPEAT FROM 1. ABOVE."
620 PRINT "TO END TEST, DEPRESS 'STOP,RUN 658, EXECUTE'."
624 PRINT "B.IF INPUT CHANGED,TEST IS ABORTED"
630 PRINT
632 PRINT "CHANGE DATA!"
633 PRINT
634 STOP
636 CMD "?W5"
638 ENTER (13,*)B
640 IF A#B THEN 644
642 GOTO 632
644 PRINT "TEST ABORTED!INPUT CHANGED TO:"B
648 PRINT "INPUT DATA CHANGED TO"B"(C)"
652 PRINT "DID YOU PUSH 'RET DATA' OR CHANGE 6940 BIT 15?"
656 GOTO 660
658 PRINT "DATA INPUT O.K.-CHECK GATE/FLAG"
659 PRINT
660 END
```

```
700 PRINT "59500/6940 GATE/FLAG VERIFICATION.FOLLOW DISPLAY INSTRUCTIONS."
702 PRINT
704 CMD "?U7","@T"
706 DISP "59500 GATE LITE ON,FLAG OFF?"
708 STOP
710 DISP "6940 LOAD OUTPUT LITE ON?"
712 STOP
714 OUTPUT (13,*)"X"
716 DISP "GATE AND LOAD OUTPUT LITES OFF?"
718 STOP
720 DISP "PUT 6940 IN LOCAL MODE (C)"
722 STOP
724 CMD "?U7","@T"
726 DISP "59500 GATE LITE ON?"
728 STOP
730 DISP "PUSH AND HOLD 6940 'RET DATA'(C)"
732 STOP
734 DISP "59500 GATE OFF,FLAG ON?"
736 STOP
738 DISP "RELEASE 6940 'RET DATA'(C)"
740 STOP
742 DISP "FLAG OFF?"
744 STOP
746 PRINT "TEST COMPLETE."
747 PRINT
748 END
```

5-52 Operating Procedures — 9820/9821 Calculator — 59500/6940 Verification

5-53 The verification program listings for a 9820/9821 calculator-based HP-IB system employing the 59500/6940 are included immediately following paragraph 5-54. To execute the programs type them in as listed in the manual, depress calculator keys END, EXECUTE, and RUN PROGRAM.

5-54 The verification programs are "stand alone" with all instructions required to operate the tests printed on the calculator printer. The programs output two types of messages, program instructions and test inquiry, that are similar to the messages output by the 9830 calculator programs and described in paragraphs 5-50 and 5-51. Note that the 9820/9821 calculator requires a different operator response than that required by the 9830. That is, instead of depressing "CONT, EXECUTE" in response to an instructon that ends in "(C)", or a test inquiry message, the 9820/9821 user should depress "RUN PROGRAM". Thus, the 9820/9821 instruction messages requiring this response all end in "(R)" instead of "(C)". Of course, like the 9830 programs, the test inquiry messages all end in a "?" and require the appropriate yes or no response as discussed in paragraph 5-51 for the 9830.

9820/9821 VERIFICATION PROGRAMS

```
0:
PRT "59500 LISTE
N","MODE","VERIF
ICATION."⊢
1:
PRT "FOLLOW PRIN
TER","INSTRUCTIO
NS"⊢
2:
SPC 2⊢
3:
PRT "TURN 59500
OFF","THEN BACK
ON(R)"⊢
4:
SPC 2⊢
5:
STP ⊢
6:
CMD "?U7"⊢
7:
PRT "LISTEN LITE
 ON?"⊢
8:
SPC 2⊢
```

```
9:
STP ⊢
10:
CMD "? !#≠$%&'()
*+-,./012345689:
;<=>"⊢
11:
FMT Y1,Z;WRT 13⊢
12:
WTB 13,34⊢
13:
FMT Y2,Z;WRT 13⊢
14:
PRT "LISTEN LITE
 OFF?"⊢
15:
SPC 2⊢
16:
STP ⊢
17:
CMD "? :!#≠$%&'(
)*+-,./012345689
:;<=>"⊢
18:
FMT Y1,Z;WRT 13⊢
19:
WTB 13,34⊢
20:
FMT Y2,Z;WRT 13⊢
21:
PRT "LISTEN LITE
 ON?"⊢
22:
SPC 2⊢
23:
STP ⊢
24:
PRT "DEPRESS STO
P","RUN PROGRAM
"⊢
25:
SPC 2⊢
26:
STP ⊢
27:
PRT "LISTEN LITE
 OFF?"⊢
28:
SPC 2⊢
29:
STP ⊢
30:
CMD "?W"⊢
31:
PRT "LISTEN LITE
 OFF","TALK LIT
E ON?"⊢
32:
SPC 2⊢
33:
STP ⊢
34:
CMD "7"⊢
35:
PRT "LISTEN LITE
 ON","TALK LITE
OFF?"⊢
36:
SPC 2⊢
37:
STP ⊢
38:
PRT "LISTEN MODE
 O.K.","CHECK TA
LK MODE."⊢
39:
SPC 8⊢
40:
END ⊢
R335
```

```
0:
PRT "TALK MODE",
"VERIFICATION"⊢
1:
PRT "FOLLOW PRIN
TER","INSTRUCTIO
NS"⊢
2:
SPC 3⊢
3:
PRT "TURN 59500"
,"OFF THEN ON(R)
"⊢
4:
SPC 2⊢
5:
STP ⊢
6:
CMD "+"⊢
7:
CMD "W"⊢
8:
PRT "59500 TALK"
,"LITE ON?"⊢
```

Talk Mode Verification Continued

```
9:
SPC 2⊢
10:
STP ⊢
11:
CMD "→"⊢
12:
PRT "TALK LITE O
FF?"⊢
13:
SPC 2⊢
14:
STP ⊢
15:
PRT "TALK LITE S
HOULD","BLINK ON
CE(R)"⊢
16:
SPC 2⊢
17:
STP ⊢
18:
64→A⊢
19:
"Q";IF A=94;GTO
"A"⊢
20:
IF A=87;GTO "B"⊢
21:
CMD "W"⊢
22:
FMT Y1,Z;WRT 13⊢
23:
WTB 13,A⊢
24:
FMT Y2,Z;WRT 13⊢
25:
PRT "TALK LITE",
"BLINK ONCE?"⊢
26:
SPC 2⊢
27:
STP ⊢
28:
"B";A+1→A⊢
29:
GTO "Q"⊢
30:
"A";CMD "W7"⊢
31:
PRT "TALK LITE O
FF","LISTEN LITE
 ON?"⊢
32:
SPC 2⊢
33:
STP ⊢
34:
CMD "W"⊢
35:
PRT "DEPRESS STO
P","RUN PROGRAM
"⊢
36:
SPC 2⊢
37:
STP ⊢
38:
PRT "TALK LITE O
FF?"⊢
39:
SPC 2⊢
40:
STP ⊢
41:
PRT "TALK MODE O
.K.","CHECK","SE
RIAL POLL."⊢
42:
SPC 8⊢
43:
END ⊢
R339
```

```
0:
PRT "SERIAL POLL
","VERIFICATION"
⊢
1:
PRT "FOLLOW PRIN
TER","INSTRUCTIO
NS."⊢
2:
SPC 3⊢
3:
PRT "TURN 59500"
,"OFF THEN ON(R)
"⊢
4:
SPC 2⊢
5:
STP ⊢
6:
CMD "?U7","00020
T"⊢
```

```
7:
PRT "59500 SERVI
CE","REQUEST ON?
";SPC 2;STP ⊢
8:
IF (RDS 13=0)+(
RDS 13=1);GTO "X
"⊢
9:
PRT "59500 SRQ",
"MALFUNCTION"⊢
10:
"A";GTO "FIN"⊢
11:
"X";CMD "?→5%"⊢
12:
CMD "W";RDB 13→R
2⊢
13:
IF R2=64;GTO "GO
OD"⊢
14:
PRT "STATUS BYTE
","ERROR","BIT
7 OFF."⊢
15:
SPC 2;GTO "FIN"⊢
16:
"GOOD";PRT "TALK
 AND","SERIAL PO
LL ON;","SERVICE
 REQ.OFF?";SPC 2
;STP ⊢
17:
CMD "?→%"⊢
18:
PRT "SERIAL POLL
 OFF?";SPC 2;
STP ⊢
19:
CMD "?→5%";CMD "
W";RDB 13→R2⊢
20:
IF R2≠64;GTO "K"
⊢
21:
PRT "STATUS BYTE
","ERROR","BIT
7 ON";SPC 2;GTO
"FIN"⊢
22:
"K";CMD "?U7","0
0020T"⊢
23:
PRT "DEPRESS 'ST
OP'","'RUN PROGR
AM'";SPC 2;STP ⊢
24:
PRT "SERVICE REQ
. ON?";SPC 2;
STP ⊢
25:
PRT "SERIAL POLL
 O.K.","CHECK","
DATA OUTPUT.";
SPC 8⊢
26:
"FIN";END ⊢
R323
```

```
0:
PRT "DATA OUTPUT
","VERIFICATION.
","FOLLOW PRINTE
R","INSTRUCTIONS
."⊢
1:
SPC 2⊢
2:
PRT "TURN 6940 O
FF.","TURN 59500
","OFF THEN ON."
,"TURN 6940 ON.(
R)"⊢
3:
SPC 2;STP ⊢
4:
CMD "?U7","07777
"⊢
5:
PRT "ALL 6940 DA
TA","LITES ON?";
SPC 2;STP ⊢
6:
CMD "?U7","@"⊢
7:
PRT "ALL 6940 DA
TA","LITES OFF?"
⊢
8:
SPC 2⊢
9:
STP ⊢
10:
CMD "?U7","M7777
"⊢
```

```
11:
PRT "ONLY 6940 B
IT"⊢
12:
PRT "13 OFF?"⊢
13:
SPC 2⊢
14:
STP ⊢
15:
CMD "?U7","N7777
"⊢
16:
PRT "ONLY 6940 B
IT"⊢
17:
PRT "12 OFF?"⊢
18:
SPC 2⊢
19:
STP ⊢
20:
11→C;3→A;2→B⊢
21:
"A";7777-2↑B*10↑
A→X⊢
22:
FMT "O",FXD *.0;
WRT 13,X⊢
23:
PRT "ONLY 6940 B
IT"⊢
24:
PRT C,"OFF?"⊢
25:
SPC 2⊢
26:
STP ⊢
27:
B-1→B;IF 0>B;
GTO "B"⊢
28:
C-1→C⊢
29:
GTO "A"⊢
30:
"B";A-1→A;2→B;
IF 0>A;GTO "C"⊢
31:
C-1→C⊢
32:
GTO "A"⊢
33:
"C";CMD "?U7","E
2525"⊢
34:
PRT "ONLY 6940 E
VEN"⊢
35:
PRT "(0,2...)BIT
S ON?"⊢
36:
SPC 2⊢
37:
STP ⊢
38:
CMD "?U7","J5252
"⊢
39:
PRT "ONLY 6940 O
DD"⊢
40:
PRT "BITS ON?"⊢
41:
SPC 2⊢
42:
STP ⊢
43:
CMD "?U7","@0
44:
CMD "?U","O7777"
⊢
45:
PRT "ALL 6940 DA
TA"⊢
46:
PRT "BITS OFF?"⊢
47:
SPC 2⊢
48:
STP ⊢
49:
CMD "?U7","O7777
"⊢
50:
CFG 1⊢
51:
"E";0→A⊢
52:
"F";WTB 13,A⊢
53:
IF A=47;GTO "G"⊢
54:
A+1→A⊢
55:
GTO "F"⊢
```

```
56:
"G";56→A⊢
57:
"H";WTB 13,A⊢
58:
IF A=62;GTO "I"⊢
59:
A+1→A⊢
60:
GTO "H"⊢
61:
"I";80→A⊢
62:
"J";WTB 13,A⊢
63:
IF A=83;GTO "L"⊢
64:
IF A=86;A+1→A;
GTO "L"⊢
65:
IF A=89;GTO "L"⊢
66:
IF A=94;GTO "L"⊢
67:
IF A>127;GTO "M"
⊢
68:
A+1→A⊢
69:
GTO "J"⊢
70:
"L";A+2→A⊢
71:
GTO "J"⊢
72:
"M";IF FLG 1=1;
GTO "N"⊢
73:
PRT "ALL 6940 DA
TA"⊢
74:
PRT "BITS ON?"⊢
75:
SPC 2⊢
76:
STP ⊢
77:
SFG 1⊢
78:
"?U7""@"⊢
79:
GTO "E"⊢
```


```
80:
"N";PRT "ALL 694
0 DATA"⊢
81:
PRT "BITS OFF?"⊢
82:
SPC 2⊢
83:
STP ⊢
84:
PRT "DATA OUTPUT
 O.K"⊢
85:
PRT "CHECK DATA"
⊢
86:
PRT "INPUT";SPC
8⊢
87:
END ⊢
R259
```

```
0:
PRT "DATA INPUT"
⊢
1:
PRT "VERIFICATIO
N"⊢
2:
PRT "FOLLOW PRIN
TER"⊢
3:
PRT "INSTRUCTION
S."⊢
4:
SPC 2⊢
5:
PRT "TURN 6940 O
FF."⊢
6:
PRT "TURN 59500"
⊢
7:
PRT "OFF THEN ON
."⊢
8:
PRT "TURN 6940 O
N.(R)"⊢
9:
SPC 2⊢
10:
STP ⊢
```

```
11:
"?U7""OT0"⊢
12:
PRT "PUT 6940 IN
 LOC."⊢
13:
PRT "WHEN ENTER"
⊢
14:
PRT "DATA! APPEA
RS"⊢
15:
PRT "INPUT DATA"
⊢
16:
PRT "FROM 6940-"
⊢
17:
PRT "1.SET BITS"
⊢
18:
PRT " 15,11-0,"⊢
19:
PRT "2.PUSH 6940
"⊢
20:
PRT " 'RETURN DA
TA',"⊢
21:
PRT "3.DEPRESS C
ALC."⊢
22:
PRT " 'RUN PROGR
AM',"⊢
23:
PRT "4.CHECK PRI
NTER"⊢
24:
PRT "A.IF INPUT
O.K."⊢
25:
PRT " GO TO STEP
 5"⊢
26:
PRT "B.IF INPUT
N.G."⊢
27:
PRT "FIX PROBLEM
"⊢
28:
PRT "5.A.TO INPU
T"⊢
```

```
29:
PRT " MORE DATA,
"⊢
30:
PRT "PRESS'RUN P
RGRM'"⊢
31:
PRT "B.TO CONTIN
UE"⊢
32:
PRT "INPUT TEST"
⊢
33:
PRT "PRESS 'GO T
O A'"⊢
34:
PRT "THEN'RUN PR
GRM'"⊢
35:
SPC 8⊢
36:
"B";PRT "ENTER D
ATA!"⊢
37:
SPC 2⊢
38:
STP ⊢
39:
CMD "?W5"⊢
40:
FMT FXD 5.0;RED
13,A⊢
41:
PRT A⊢
42:
SPC 2⊢
43:
STP ⊢
44:
GTO "B"⊢
45:
"A";PRT "SET 694
0 DATA"⊢
46:
PRT "TO 7777,"⊢
47:
PRT "PUSH'RET DA
TA',"⊢
48:
PRT "THEN'RUN PR
GRM'"⊢
49:
SPC 2⊢
50:
STP ⊢
```

```
51:
CMD "?W5"⊢
52:
FMT FXD 5.0;RED
13,A⊢
53:
PRT "WHEN CHANGE
"⊢
54:
PRT "DATA! APPEA
RS-"⊢
55:
PRT "1.SET 6940
BITS"⊢
56:
PRT "11-0,"⊢
57:
PRT "2.DON'T PRE
SS"⊢
58:
PRT "'RET DATA'"
⊢
59:
PRT "3.PRESS'RUN
 PRG'"⊢
60:
PRT "4.CHANGE DA
TA"⊢
61:
PRT "AS DESIRED"
⊢
62:
PRT "TO END TEST
-"⊢
63:
PRT "PRESS'GOTO
D'"⊢
64:
PRT "THEN'RUN PR
GRM'"⊢
65:
SPC 8⊢
66:
"C";PRT "CHANGE
DATA!"⊢
67:
SPC 2⊢
68:
STP ⊢
69:
CMD "?W5"⊢
70:
FMT FXD 5.0;RED
13,B⊢
71:
IF A≠B;GTO "F"⊢
72:
GTO "C"⊢
73:
"F";PRT "TEST AB
ORTED!";PRT "INP
UT CHANGED TO";
PRT B⊢
74:
PRT "DID YOU PUS
H"⊢
75:
PRT "'RET DATA'?
"⊢
76:
PRT "OR CHANGE B
IT 15?"⊢
77:
SPC 2;GTO "E"⊢
78:
"D";PRT "DATA IN
PUT O.K."⊢
79:
PRT "CHECK GATE/
FLAG";SPC 8⊢
80:
"E";END ⊢
R257
```

```
0:
PRT "GATE/FLAG"⊢
1:
PRT "VERIFICATIO
N"⊢
2:
PRT "FOLLOW PRIN
TER"⊢
3:
PRT "INSTRUCTION
S."⊢
4:
SPC 2⊢
5:
PRT "TURN 6940 O
FF.";PRT "TURN 5
9500"⊢
```

```
6:
PRT "OFF THEN ON
.";PRT "TURN 694
0 ON.(R)"⊢
7:
SPC 2⊢
8:
STP ⊢
9:
CMD "?U7","@T"⊢
10:
PRT "59500 GATE
ON,";PRT "FLAG O
FF?"⊢
11:
STP ;SPC 2⊢
12:
PRT "6940 LOAD O
UTPUT";PRT "LITE
 ON?";STP ;SPC 2
⊢
13:
FMT Z,"X";WRT 13
⊢
14:
PRT "59500 GATE
OFF?";PRT "6940
LOAD";PRT "OUTPU
T OFF?";SPC 2;
STP ⊢
```

```
15:
PRT "PUT 6940 IN
";PRT "LOCAL MOD
E(R)";SPC 2;STP
⊢
16:
CMD "?U7","@T"⊢
17:
PRT "59500 GATE
ON?";SPC 2;STP ⊢
18:
PRT "PUSH AND HO
LD";PRT "6940'RE
T DATA'(R)";SPC
2;STP ⊢
19:
PRT "59500 GATE
OFF,";PRT "FLAG
ON?";SPC 2;STP ⊢
20:
PRT "RELEASE";
PRT "'RET DATA'(
R)";SPC 2;STP ⊢
21:
PRT "59500 FLAG
OFF?";SPC 2;STP
⊢
22:
PRT "TEST COMPLE
TE.";SPC 8⊢
23:
END ⊢
R338
```

Chapter VI
PLUG-IN CARD DESCRIPTIONS AND PROGRAMS

6-1 This chapter describes the programming sequences associated with the various plug-in cards of the multiprogrammer system. The chapter is subdivided into three major sections; general card programming information, individual card programs, and multiple card programming of special purpose applications. As a prelude to this chapter, be sure to read how to program the basic word formats of the multiprogrammer system (ChapterV, Paragraphs 5-4 through 5-31) and all of Chapter II (particularly the programming sequences described in Paragraphs 2-40 through 2-65).

6-2 GENERAL CARD PROGRAMMING INFORMATION

6-3 Programming Aids

6-4 Utility Subroutines. Appendix A includes various subroutines that the user will find helpful when writing his programs. These subroutines are used throughout the programming examples in this Chapter and one of the routines (serial poll) was used previously in Example 7 of Chapter V. Each subroutine is designated by a number (e. g., 2000) in the 9830A programs, or a name (e. g., "SPOLL") in 9820/21A programs. Variables "A" and "A1" through "A4" are used in the 9830A subroutines while "A" and "R1" through "R4" are used in the 9820/21A subroutines. Of course, if any of these subroutines are utilized, the user must ensure that the applicable variables are not redefined elsewhere in his program. Each subroutine, together with a brief description is listed below:

1. Serial Poll; subroutine 2000 ("SPOLL"). This poll routine reads a status byte into variable A4 (or R4) which indicates whether or not the multiprogrammer system had requested service (set SRQ line). This subroutine also clears the SRQ line before returning to the main program.

2. Octal to Decimal Conversion; subroutine 2200 ("DEC"). This is an octal to decimal conversion routine which is useful when programming input cards. It converts octal numbers from entry variable "A" into decimal equivalents and returns to the main program with variable "A2" (9830A) or "R2" (9820/21A). Note: When using 9830A calculators with a 11279B Advanced Programming ROM, this subroutine is unnecessary and can be replaced by the statement "A2 = OCTA".

3. Decimal to Octal Conversion; subroutine 2400 ("OCT"). This decimal to octal subroutine can be used when programming output cards. It accepts decimal numbers between −2048 and +4095 and converts them into octal equivalents. The subroutine uses entry and exit variables "A" and "A2" ("R2"), respectively.

4. Decimal to Bit Conversion; subroutine 2600 ("BIT"). This subroutine is used in the sample test programs for various input cards in the individual card portion of this Chapter. It accepts a decimal variable ("A2" or "R2") and enters the bit and corresponding pin number on the calculator display.

5. Decimal to BCD Conversion; subroutine 2800 ("D-BCD"). This subroutine is convenient when programming a digital output card (69331A) that controls an external device requiring a BCD input. The subroutine converts decimal numbers, from 000 to 999, to BCD weighted octal values.

6. BCD to Decimal Conversion; subroutine 3000 ("BCD-D"). This is the inverse operation of the preceding subroutine and is helpful when programming a digital input card (69431A) that receives BCD data from its external device.

6-5 Number System Theory. Appendix A also contains a description of the number systems applicable to the multiprogrammer system. Included are explanations of the decimal, octal, binary, and BCD systems, how to convert numbers from one system to another, and how the number systems relate to some of the plug-in cards.

6-6 Control Mode Reference Table. Table 6-1 lists all of the plug-in cards available at the time of this writing and which operating mode bits (in a control word) will affect the operation of each card. Also included is an IRQ column which indicates the cards that contain IRQ circuitry.

6-7 The following two examples show how to use Table 6-1:

1. If the user has a 69321B D/A output card, he can see at a glance that it will be affected by the DTE and SYE control mode bits. He can also determine that if TME is on (extended handshake), it will require 30μsec (nominal) for the card to return a CTF flag.

2. If the user has a 69431A digital input card, he can deduce that ISL must be on to read-in data from the card and that the card has interrupt capability (IEN and IRQ). The user can also determine from the table that this card is capable of providing a gate/flag data transfer cycle with its external device when TME is programmed on. In this case, the length of the flag period is, of course, determined by the external device (Note 1 of table).

6-8 Status Checks and Serial Polls

6-9 Bus status checks and serial poll routines appear throughout the remainder of this chapter in both the output and input card programs. These procedures are employed during programs where the multiprogrammer is used in the timing mode (TME on) and can, therefore, issue a service request. A serial poll routine should also be done at the start of any program to ensure that the service request line is cleared at this point. The status check/serial poll technique is described in Chapter V (Paragraphs 5-29 through 5-31) and will not be re-explained each time it appears in this Chapter.

6-10 Output Cards, General Programming Techniques

6-11 Three multiprogrammer control mode bits (SYE, DTE, and TME) are used when programming output cards.

Table 6-1. Control Mode Summary for Plug-In Cards

MODEL	DESCRIPTION	IEN	ISL	DTE	SYE	Flag (CTF) From Card (4)	IRQ
69321B	D/A Voltage	–	–	Yes	Yes	30μsec	–
69325A	BPSA Voltage Cont.	–	–	–	Yes	9.5msec	–
69326A	BPSA Current Cont.	–	Yes	–	Yes	4.3msec	–
69327A	BPSA Current Cont.	–	Yes	–	Yes	4.3msec	–
69328A	BPSA Gain Cont.	–	–	–	Yes	9.5msec	–
69330A	Relay Output	–	–	Yes	Yes	12msec (1)	–
69331A	Digital Output	–	–	Yes	Yes	(1)	–
69332A	Open Collector Output	–	–	–	–	(5)	–
69335A	Stepping Motor	–	–	–	Yes	(2)	–
69370A	D/A Current	–	–	Yes	Yes	30μsec	–
69380A	Breadboard Output	–	–	–	–	(5)	–
69421A	Voltage Monitor	–	Yes	–	–	6msec	–
69430A	Iso. Digital Input	–	Yes	–	–	(5)	–
69431A	Digital Input	Yes	Yes	–	–	(1)	Yes
69433A	Relay Output/Readback	–	Yes	–	Yes	4msec	–
69434A	Event Sense	Yes	Yes	–	–	(3)	Yes
69435A	Pulse Counter	–	Yes	–	–	(5)	–
69436A	Process Interrupt	Yes	Yes	–	–	(3)	Yes
69480A	Breadboard Input	–	Yes	–	–	(5)	–
69500-06A	12 Bit Resistance	–	–	–	Yes	6msec	–
69510-13A	Dual 6 Bit Resistance	–	–	–	–	6msec	–
69600A	Programmable Timer	Yes	Yes	–	–	(2)	Yes
69601A (Note 6)	Frequency Reference	–	–	–	–	(5)	–

(1) Max time depends on flag from external device.
(2) Time depends on number of pulses and pulse frequency.
(3) Max time depends on external inputs to detectors.
(4) CTF flag is ignored unless TME has been programmed on.
(5) These cards have no CTF circuitry and must not be sent a data or address word with a gate ("T") while TME is on. If this is done, no flag will be returned from the card causing the system to hang-up.
(6) Free-running frequency generator (not programmable).

Table 6-1 lists the cards affected by these bits and Table 5-1 (Chapter V) shows the four control mode codes most often used for output card programming. Explanations of sending constants and variables to output cards as well as descriptions of each control mode (SYE, DTE, TME) are given in the following paragraphs. Programming examples for each type of calculator are included where applicable.

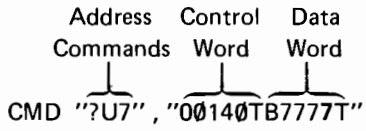
NOTE

A unit address of ØØ is used in all control words shown in the programming examples.

6-12 **Sending Constants and Variables.** As a rule, output cards are programmed with control words and data words. (There are some output cards that can also be operated in the input mode as will be pointed out in the individual card programming section.) The data field portion of a data word may contain a constant (explicit value) or a variable (value stored in calculator). Either command statements or output/write statements are used to send data to an output card. Command statements can only be used to send constants. Output (9830A) or write (9820/21A) statements can also be used to send constants but *must* be used to send variables.

6-13 Example 1 uses a command statement to send a data word containing a constant data value to an output card. In this example, a data value of "7777" is sent to the output card in slot 402 ("B").

Example 1. Sending a Constant Data Value:



6-14 Example 2 shows how an Output or Write statement is used to send a variable data value to an output card. In the example, a command statement includes the same address commands and control word as in Example 1. However, the variable data that will be transmitted to the output card is included in the output (9830) or write (9820/21) statements. Variable A2 (9830) or R2 (9820/21) can be any positive octal value between "ØØØØ" and "7777". Note that if the variable were equal to "7777", Example 2 would perform the same operation as Example 1.

6-15 Line 1ØØ in Example 2 allows the user to work with decimal numbers when programming. The 24ØØ ("OCT") subroutine converts decimal numbers between −2Ø48 and +4Ø95 to octal equivalents. If decimal to octal conversion is not desired, line 1ØØ is omitted from the program.

6-16 Note also in Example 2 that a semi-colon is used at the end of the 9830 output statement, and Format Z is used in the 9820/21 write statement. The use of the semi-colon or Format Z suppresses the carriage return/line feed codes at the end of the output or write statement. Suppression of the carriage return/line feed codes is essential when outputting variables when the multiprogrammer is in the timing mode. The reason for this is that the calculator must receive handshake flags with the carriage return/line feed codes in order to continue with the program. Thus, by suppressing the carriage return/line feed codes, the calculator can continue to the next statement in the program while the multiprogrammer is still timing out. Note that all of the output/write statements shown in this chapter will use the semi-colon/FMT Z; even those used in a non-timing mode (TME off). It does no harm to do this, and it helps avoid potential hang-up problems.

6-17 **System Enable (SYE).** The SYE control mode is used to control the outputs of most output cards (see Table 6-1). To enable the outputs, SYE must be programmed on. When SYE is programmed off, the outputs of all cards affected by SYE are disabled; i. e., are placed in a "safe" condition. For example; resistance outputs are shorted, voltage outputs are held at 0 volts and digital outputs are held in the open or zero state.

6-18 As shown in previous examples, SYE can be

Example 2. Sending a Variable Data Value:

9830A Calculator

```
1ØØ GOSUB 24ØØ
11Ø CMD "?U7" , "ØØ14ØT"
12Ø FORMAT F1ØØ5.Ø
13Ø OUTPUT (13, 12Ø) "B"A2"T";
```

9820/21A Calculators

```
GSB "OCT"
CMD "?U7" , "ØØ14ØT"
FMT Z, "B", FXD * .Ø; WRT 13, R2
FMT Z, "T", FXD * .Ø; WRT 13
```

Explanation:

11Ø Control word is sent to multiprogrammer (DTE, SYE on).
12Ø-13Ø: Variable data word (formatted as specified) is sent to output card in slot 402 ("B"), with gate ("T").

programmed on in a control word prior to sending the data word to the output card. In some instances, the user may wish to take advantage of the SYE off condition which removes the outputs from all affected output cards simultaneously. In this case, transmitting a control word with SYE off will cause all affected cards to go to the "safe" condition. Example 3 shows a control word with SYE off. Since leading zeros need not be sent, the control word consists only of "O" (control word tag) and "T" (multiprogrammer gate code). When Example 3 is executed, all output cards responding to SYE will go to a "safe" condition.

Example 3. Transmitting a Control Word With SYE Off:

CMD "?U7", "OT"

6-19 Thus, all output cards affected by SYE are in a "safe" condition when SYE is off and will output data when SYE is on, whether or not the card is addressed.

6-20 Data Transfer Enable (DTE). The DTE control mode affects output cards having either external gate/flag capability or dual rank storage. DTE need not be programmed on if a particular card does not have either of these capabilities, but no harm is done if it is programmed on. When using the timing mode, it is generally good practice to program both TME and DTE on together. The reason for this is that the four cards affected by DTE (69321B, 69330A, 69331A, and 69370A) cannot drive the CTF flag line unless DTE is on. Thus, is any of these cards are used in the timing mode, with DTE off, a hang-up will occur. When DTE is on and one of the affected cards is addressed and gated ("T" character) it will immediately return the leading edge of the CTF flag. Depending upon the card model, the trailing edge of the flag is either controlled by circuits on the card itself or by an external circuit connected to the card. For models 69321B and 69370A (D/A cards), the trailing edge of the flag is controlled by circuits on the card itself. For models 69330A (Relay Output) and 69331A (Digital Output), the trailing edge of the flag is normally controlled by an external device.

6-21 Cards With External Gate/Flag Capability. The digital output card (69331A) and the relay output card (69330A) have external gate/flag capability. This feature allows these cards to transfer data to an external device which requires a gate input to receive data.

6-22 Normally DTE will be programmed on in a control word, before sending a data word to the card, as shown in the Examples 1 and 2. As each data word is received, it is loaded into the addressed card and immediately strobed (gated) into an external device.

6-23 Some multiple card applications may require data to be loaded into several output cards, then simultaneously gated into external devices. In this case, a control word may be sent with DTE off, followed by appropriate data words, then another control word with DTE on. As each data word is received, the data will be immediately transferred to the output of the addressed card. However, the gate from the output card will not be enabled until a control word containing DTE is received.

6-24 Example 4 illustrates loading data into several output cards with DTE off, then turning DTE on to simultaneously generate gates to external devices.

Example 4. Programming Multiple Cards With DTE:

CMD "?U7", "00040TA7777TB7777TC7777T00140T"

6-25 In Example 4, output cards located in multiprogrammer slots 401, 402, and 403 (A, B, C) are sequentially loaded with data. Notice that SYE is on, allowing the data to be immediately transferred to the output of the cards. After all cards have been loaded with data, DTE is turned on by the control word, "00140T", resulting in simultaneous generation of gate signals to external devices.

6-26 If it is not required to simultaneously gate external devices from multiple cards, DTE can be included in the first control word of Example 4, (i. e., 00140T) and the control word following the data words can be eliminated. Each card would then generate a gate as it receives a data word, as previously described.

6-27 Cards With Dual Rank Storage. The D/A voltage (69321B) and current (69370A) converter output cards have dual rank storage capability. This feature allows the outputs of any number of D/A cards to be changed to new values simultaneously. In this case, a control word is sent with DTE off, followed by appropriate data words, then another control word with DTE on. As each data word is received, the data will be loaded into the first rank storage of the addressed card. When a subsequent control word with DTE is received, all previously addressed D/A cards will simultaneously transfer the contents of the first rank storage into second rank storage and to the output D/A conversion circuits.

6-28 Example 4, which was previously shown for programming DTE for digital output and relay output cards, can also be used for D/A cards. In this case, the first rank storage of D/A cards located in multiprogrammer slots 401, 402, and 403 (A, B, C) would be loaded with new data. After all cards have received new data, DTE is turned on by the control word "00140T", resulting in all three cards transferring data from first rank to second rank storage and converting the data to analog outputs.

6-29 As was the case for the 69330A and 69331A cards, DTE can also be programmed on first, before sending data words to the D/A cards. As each data word is received, it will be loaded into the addressed card and converted to an analog voltage/current that will be present at the output of the card.

6-30 Timing Mode (Extended Handshake). The TME control mode allows output cards with CTF circuits to control the multiprogrammer flag. Table 6-1 lists the cards which may be used in the TME mode. When programming output cards with TME on, the receipt of a CTF card flag will set the service request (SRQ) bus line to notify the calculator that the output card(s) has timed out (completed data transfer). A status check and serial poll is then performed to confirm that the multiprogrammer system had requested service and to reset the SRQ line.

6-31 Serial and parallel methods may be used when programming output cards in the timing mode. The serial method programs one output card at a time, while the parallel method involves programming several output cards at once.

6-32 Serial Method. Example 5 illustrates programming output cards serially (one at a time) in the timing mode. Two output cards are utilized in this example. A control word with both TME and DTE on is transmitted first placing the multiprogrammer in timing mode. Since certain output cards cannot drive CTF unless DTE is on, DTE and TME should be programmed on together (see Paragraph 6-20). With TME on, it is then necessary to jump to a serial poll routine before addressing the first output card in a data word. This clears the service request line, which was set by

Example 5. Timing Mode of Operation, Serial Method:

9830 Calculator	9820/21 Calculators
110 CMD "?U7", "00160T"	CMD "?U7", "00160T"
120 GOSUB 2000	GSB "SPOLL"
130 CMD "?U7", "B7777T"	CMD "?U7", "B7777T"
.	.
.	.
.	.
200 IF STAT 13 > 1 THEN 500	IF RDS 13 > 1; GTO "CONT"
210 GOSUB 2000	GSB "SPOLL"
220 IF A4#64 THEN 500	IF R4≠64; GTO "CONT"
230 CMD "?U7", "C7777T"	CMD "?U7", "C7777T"
.	.
.	.
.	.
500 REM CONTINUE PROGRAM	"CONT" CONTINUE PROGRAM

Explanation:

110 Control word is sent to multiprogrammer with DTE, SYE, and TME on.
120 Serial Poll routine resets service request caused by control word with TME on.
130 Data word is sent to the output card in multiprogrammer slot 402 ("B"). Gate character ("T") initiates timing sequence with first card.
200 Bus status check; if service not requested go to continue main program, line 500.
210 Service requested; go to serial poll subroutine for multiprogrammer.
220 Examine MP status byte in register; if not equal to 64, service not requested by MP; go to continue main program, line 500.
230 MP requested service and data transfer with first card is complete. Data word is sent to the output card in multiprogrammer slot 403 ("C") to initiate timing sequence with second card. Status check and serial poll for second card would follow this line.

the trailing edge of the CTF handshake flag when the control word with TME was transmitted. From this point on, the service request will be set only as a function of an output card timing out.

6-33 When the first card times out, the service request (SRQ) line is set. The calculator identifies the source of the service request by a status check of the bus and subsequent serial poll. Thus informed that the first card has timed out, a data word is sent to initiate a timing operation with the second card. The same sequence of status check and serial poll must be used to determine completion of the timing operation with the second card. Notice that the main program may have to loop back to the status check to determine when a card times out.

6-34 Parallel Method. Example 6 illustrates parallel (simultaneous) programming of two output cards in the timing mode. A control word with DTE and SYE, followed by data words addressing the two cards, and then another control word with DTE, SYE, and TME are sent to the multiprogrammer. Each output card will begin its timing sequence upon receipt of the "T" in its data word. Since TME is turned on after the timing sequences have started, a trailing edge of the CTF flag will not be returned to set the multiprogrammer's service request until *both* cards have timed out.

6-35 If a status check of the bus and subsequent serial poll indicate that the multiprogrammer system is not requesting service (one or both output cards have not timed out), the program is allowed to continue as the user desires. If, however, a status check and subsequent serial poll indicate that the multiprogrammer system is requesting service (both cards have timed out), the user can take appropriate action; e. g., send new data to the output cards.

6-36 When programming one output card in the timing mode, the user has the option of using either the serial or the parallel method. The user will probably find the parallel method more convenient since it eliminates the necessity for the initial serial poll required in the serial method.

Example 6. Timing Mode of Operation, Parallel Method:

9830 Calculator

```
110 CMD "?U7", "00140TA7777TB7777T00160T"
                    .
                    .
                    .
200 IF STAT 13 > 1 THEN 500
210 GOSUB 2000
220 IF A4#64 THEN 500
230 REM ALL DATA TRANSFERS ARE COMPLETE
                    .
                    .
                    .
500 REM CONTINUE PROGRAM
```

9820/21 Calculators

```
CMD "?U7", "00140TA7777TB7777T00160T"
                    .
                    .
                    .
IF RDS 13 > 1; GTO "CONT"
GSB "SPOLL"
IF R4≠64; GTO "CONT"
   (ALL DATA TRANSFERS ARE COMPLETE)
                    .
                    .
                    .
"CONT" CONTINUE PROGRAM
```

Explanation:

110 Control word with DTE, SYE on is sent to MP. Data words are sent to output cards in slots 401 ("A") and 402 ("B"). The "T" in each data word initiates the timing sequence for each card. Another control word with DTE, SYE, and TME on is sent to MP. (With TME on the service request will not be set until both cards time out).
200 Bus Status Check, if service not requested, go to continue main program, line 500.
210 Service requested; go to serial poll routine for multiprogrammer.
220 Examine MP status byte in register, if not equal to 64, service not requested, go to continue program, line 500.
230 Multiprogrammer service request (both cards timed out), take appropriate action.

6-37 Avoiding Hang-Up Conditions in the Timing Mode. During the TME mode, there is a wait-for-flag interval that starts from the time that the output card is addressed with a gate ("T") and ends when the card returns a CTF flag and thus sets the service request line. During the wait-for-flag period, the calculator is said to "hang-up" if another output or input operation is attempted. That is, any output words sent to the multiprogrammer system will not be obeyed. If the calculator is in hang-up because of a user error, it can remain there indefinately; waiting for a service request (CTF flag) that will never occur. For output card programming in the TME mode, hang-up can be caused by one of three possible problems:

(1) A card without CTF circuitry (Note 5, Table 6-1) was addressed and gated (with a data word) while TME was on.

(2) A card with external gate/flag capability (69330A or 69331A) was addressed and gated, as is normal, *but* the gate/flag terminals on the card are open (CTF flag cannot be terminated).

(3) DTE was not turned on with TME and a card affected by DTE (Table 6-1) was addressed and gated.

6-38 The hang-up of problem (1) results in the initiation of a gate within the multiprogrammer which cannot be terminated. This hang-up can be cleared by sending an "X" prior to the desired control word, as shown in Example 10 later in this chapter. To avoid this problem, the user must bear in mind that cards without CTF circuits are not intended for timing mode operation and should never be addressed and gated while TME is on. Problem (2) of the previous paragraph results in an unterminated CTF flag and cannot be cleared by sending the statement of Example 10. The only way to clear this hang-up is to remove input power from the 6940B unit (toggle the LINE switch off and then on). To avoid this problem, the user must simply remember to jumper the gate and flag terminals together on any 69330A or 69331A cards that do not use an external timing circuit. Problem number (3) results in an unterminated gate within the multiprogrammer, the same as problem (1), and can be cleared by sending an "X" as indicated in Example 10. As described in Paragraph 6-20, problem (3) can be avoided by programming TME and DTE on together.

6-39 Input Cards, General Programming Techniques

6-40 Control Mode Bits for Input Card Operations. As indicated in Table 6-1, three control mode bits are applicable to input card programming ISL, TME, and IEN. The input select (ISL) bit must be programmed on prior to any basic read sequence in which return data is sent back to the calculator from an input card. The TME and IEN bits are used to establish the timing and interrupt modes of operation, as will be explained subsequently. Table 5-1 (Chapter V) lists the three control mode codes that are most commonly used for input card operations.

NOTE

All control words used in this section assume a unit address of 00.

6-41 System Enable Bit. Notice that the SYE bit is on in all three of the input mode codes shown in Table 5-1. Although SYE does not affect input cards, it is usually good practice to have SYE on in control words specifying an input mode of operation. If SYE is turned off, and output cards are included in your system, the outputs of these cards will be disabled creating what may be an undesirable situation. Hence, all of the control word examples shown in the input portion of this chapter show SYE on.

6-42 Basic Input Operating Modes. As shown previously in Figure 2-12 of Chapter II, there are two basic programming modes for input cards; serial and parallel. A serial operation can involve the programming of a single input card or a group of cards, programmed one at a time. A parallel operation, involves the simultaneous programming of a group of cards.

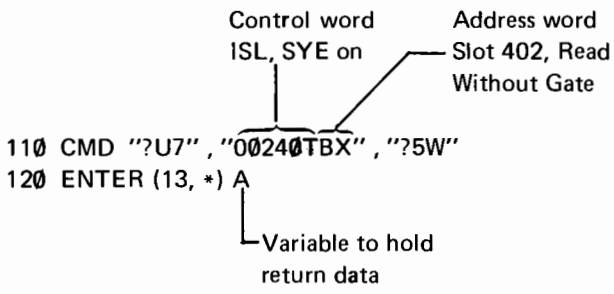
6-43 Within the boundaries of serial operation there are two programming sequences which are often used with input cards. These are the simple direct read without gate sequence and the timing mode (extended handshake). In the direct read sequence the input card is addressed, without a gate, and the existing data on the card is returned immediately to the calculator. In the timing mode sequence (TME on) the card is addressed with a gate which initiates a gate/flag transfer cycle between the card and its' external device. After the transfer is completed, the fresh data on the input card is read into the calculator.

6-44 Parallel operation for input cards is synonymous with the interrupt mode of operation (TME and IEN on). The interrupt mode allows the user to activate (arm) a group of input cards and then provides an indication (in the form of a service request) when the first card is ready with data or the first external process has been completed. Sample programs for the read without gate sequence, timing mode and interrupt mode are provided in the following paragraphs.

6-45 Reading in a Return Data Word. At some point(s) in any input card program, the user will want to read the return data available on the card into the calculator. An example of a simple read without gate sequence was shown in Example 6 of Chapter V and described in Paragraph 5-28. Because this operation is used many times when programming

Example 7. Addressing an Input Card and Entering Its Return Data:

9830 Calculator



9820/21 Calculators

```
CMD "?U7", "00240TBX", "?5W"
FMT *; RED 13, A
```

input cards, the example of the previous chapter is shown again for convenience. To summarize the read without gate sequence shown in Example 7, a control word is sent with ISL on, followed by an address word directing that the input card in slot 402 return its' data. The ENTER (or READ) statement allows the return data to be stored in the variable "A". Notice that the read without gate sequence of Example 7 is the only recommended way to obtain data from input cards that do not have CTF flag circuits (see Note 5 of Table 6-1). If the user desires to employ a direct read without gate sequence for a second input card, he can simply follow line 120 of Example 7 with an address word for the different input card slot, and then send another enter (read) statement using a different variable. This procedure can then be repeated for any additional input cards. Of course, the user must remember to include the appropriate address commands ("U7" and "5W") before and after each address word.

6-46 Checking for IRQ in the Return Data Word. As indicated in Chapter V, the return data stored in the variable of Example 7 can be used directly if it is obtained from an input card that does not contain IRQ (see Table 6-1). If the card does contain the IRQ function, the return data must first be examined to determine if there is a "1" in the IRQ field. If the IRQ character is a "1", it must be subtracted from the return data before the data can be utilized. Example 8 shows how to check for IRQ and subtract it, if necessary. The two statements of lines 130 and 140 should follow the ENTER statement of Example 7 for any cards that have IRQ. In line 130, the return data in variable "A" is checked for the presence of IRQ. If IRQ is present, the number 10000 is subtracted (in octal) from the return data in "A"

6-47 Using the Octal to Decimal Subroutine. For some input cards, such as the 69431A Digital Input, the user will find it convenient to work with the octal data returned to the variable "A" in the previous examples. For other cards, such as the 69421A Voltage Monitor, it will be more convenient to work with decimal values. In these cases, the octal to decimal subroutine (2200 or "DEC") can be utilized. To go to this subroutine, simply insert the following statement in your program: GOSUB 2200 (9830A) or GSB "DEC" (9820/21A). The placement of this program line depends on whether or not the card used IRQ. If IRQ is not used, it can be inserted immediately after line 120 of Example 7. If IRQ is used, the line must be inserted *after* the octal subtraction of IRQ (line 140 of Example 8).

6-48 Timing Mode (Extended Handshake). The TME control mode permits a data input transfer cycle between an input card and its external device. This is accomplished via the gate/flag method in which input cards with CTF circuits control the flag line. (Table 6-1 shows the input cards that have CTF circuits.) During the TME mode, the trailing edge of the CTF flag will set the service request (SRQ) line to notify the calculator that the input card has completed its data transfer. A status check and serial poll is then done to confirm that the multiprogrammer had requested service and to reset the SRQ line.

6-49 Example 9 shows how to program a single input card in the timing mode. If more than one card is involved,

Example 8. Checking for IRQ and Subtracting it:

9830 Calculator

```
130 IF A < 10000 THEN 150
140 A = A – 10000
150 REM CONTINUE PROGRAM
```

9820/21 Calculators

```
IF A ≤ 9999; GTO "CONT"
A – 10000→A
"CONT" CONTINUE PROGRAM
```

Example 9. Timing Mode for One Input Card:

9830 Calculator

```
110 CMD "?U7", "00260T"
120 GOSUB 2000
130 CMD "?U7", "BT"
              .
              .
              .
200 IF STAT 13 > 1 THEN 500
210 GOSUB 2000
220 IF A4#64 THEN 500
230 CMD "?5W"
240 ENTER (13, *) A
250 IF A < 10000 THEN 500
260 A = A - 10000
              .
              .
              .
500 REM CONTINUE PROGRAM
```

9820/21 Calculators

```
CMD "?U7", "00260T"
GSB "SPOLL"
CMD "?U7", "BT"
              .
              .
              .
IF RDS 13 > 1; GTO "CONT"
GSB "SPOLL"
IF R4≠64; GTO "CONT"
CMD "?5W"
FMT *; RED 13, A
IF A ≤ 9999; GTO "CONT"
A - 10000→A
              .
              .
              .
"CONT" CONTINUE PROGRAM
```

Explanation:

110 Control word sent with ISL, SYE, and TME on.
120 Serial poll routine resets service request caused by control word with TME on.
130 Address word sent to input card in slot 402 ("B"). Gate character ("T") initiates data transfer sequence between card and external device.
200 Bus status check; if service not requested go to continue main program, line 500.
210 Service requested; go to serial poll subroutine for multiprogrammer.
220 Examine MP status byte in register; if not equal to 64, service not requested by MP; go to continue main program. line 500.
230 MP requested service, indicating data transfer to card is complete. Place multiprogrammer in talk mode.
240 Enter data from input card into calculator variable "A".
250-260 Check data for IRQ and subtract, if necessary. (These two statements can be eliminated if card does not use IRQ.)

a serial operation can be done by simply repeating the procedures in Example 9 for each additional card.

6-50 Notice that in line 120 of Example 9, it is necessary to go to a serial poll subroutine before addressing the input card. This clears the SRQ line which was set by the CTF flag when the control word with TME on was received. From this point on, SRQ will be set only when the input card data transfer is complete. Notice that if service is not requested at line 200, the main program would have to loop back to this status check at a later time.

6-51 Avoiding "Hang-up" Conditions in the Timing Mode. During the TME mode, there is a wait-for-flag interval that starts from the time that the input card is addressed with a gate ("T") and ends when the card returns a CTF flag and thus sets the service request line. During the wait-for-flag period, the calculator is said to "hang-up" if another input or output operation is attempted. That is, any output words sent to the multiprogrammer system will not be obeyed. If the calculator is in hang-up because of a user error, it can remain there indefinitely; waiting for a service request (CTF flag) that will never occur. For input card programming in the TME mode, hang-up can be caused by one of two possible problems.

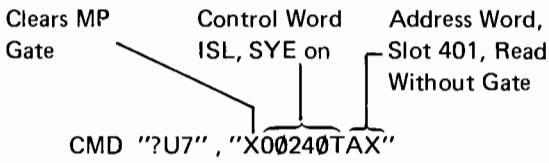
(1) A card without CTF circuitry (see Note 5, Table 6-1) was addressed and gated ("T" sent in an address word) while TME was on.

(2) A card with external gate/flag capability (69431A only) was addressed and gated, as is normal, but the gate/flag terminals on the card are open (CTF flag cannot be

terminated).

6-52 The hang-up of problem number (1) results in the initiation of a gate within the multiprogrammer which cannot be terminated. Cards without CTF circuits are not intended for timing mode operation and should never be addressed and gated while TME is on. Data from these cards should be obtained only by reading without a gate, as previously recommended in Paragraph 6-45. If you think that you have improperly gated one of these cards, the following example shows a method of clearing the hang-up condition.

Example 10. Clearing a Hang-Up Caused by an Unterminated Gate:



The "X" before the control word of Example 10 clears multiprogrammer system by terminating the gate signal, allowing a control word to be sent. The control word turns ISL on and the address word ("AX") initiates a read without gate sequence for the input card in slot 401 which is assumed to have been improperly addressed. Problem number (2) of the previous paragraph is caused by an unterminated CTF flag and cannot be cleared by sending the statement of Example 10. The only remedy for an unterminated flag is to remove input power from the 6940B unit (toggle the LINE switch off and then on). To avoid this problem the user must simply remember to jumper the gate and flag terminals together on any 69431A cards that do not use an external timing circuit.

6-53 **Interrupt Mode of Operation.** As indicated in Table 6-1, the interrupt mode can be used only with 69431A, 69434A, 69436A, or 69600A cards. These cards must be activated (armed) before they are able to cause an interrupt. As described in the interrupt mode description of Chapter II, (Paragraphs 2-52 through 2-59), there are two ways of arming the cards; individually or as a group. The most commonly used method is individual arming which is permitted only when jumper W6 is removed from the card. For this method, a control word is sent first, with ISL on. The cards are then armed, one at a time, by sending each card an address word with a gate ("T"). The interrupt mode is then established by sending a control word with TME and IEN on. The first armed card that interrupts (receives a data ready indication from its external device), returns a CTF flag setting the service request line. After a status check and serial poll, TME and IEN are turned off to prevent further interrupts and an IRQ poll (interrupt search) is performed. During the IRQ poll, a read without gate sequence is done for each card and the return data is examined for the presence of IRQ. If the IRQ character is a "1", it indicates that the addressed card had interrupted and its' return data can be stored for processing. If the IRQ character is a "0", it indicates that the card has not yet received new data or detected a change of state in the external device. Once all the cards have been polled, the interrupting card (or cards) must be reset (either disarmed or recycled) or it will cause another interrupt if TME and IEN are programmed on again. As indicated in Table 6-2, the requirements for recycling, disarming, and interrupting are different for each card. Notice also, that most of these cards are recycled or disarmed in the output mode using data words. Interrupt mode programs for these cards are given in the individual card programs later in this chapter. Additional information can be found in the instruction manual for each of these cards.

6-54 An interrupt sequence using the group arming (jumper W6 installed) technique is similar to the procedures just described except for the arming procedures at the beginning of the program. As indicated in Figure 2-12, all cards with jumper W6 installed are armed simultaneously when a control word is sent with TME and IEN on. Once an interrupt occurs (SRQ line set), an IRQ poll is done in a manner similar to that described previously in the individual arming sequence. Although the group arming technique saves programming steps, it can be difficult to use unless all input cards in the system are being used in the same manner and the user fully understands all of the implications (read paragraph 2-59 in Chapter II).

6-55 Example 11 illustrates an interrupt sequence using three input cards from which jumper W6 has been removed (cards are individually armed). The example assumes that the input cards are located in slots 401, 402, and 403 (A, B, and C).

6-56 At the completion of the interrupt sequence of Example 11, the user may wish to begin a non-interrupt operation. In this case, it is a good practice to disarm any input cards that are still active (had not interrupted) in order to avoid any possibilities of hang-ups or erroneous results in the other modes. Similarly, the user should include only the four card types listed in Table 6-2 in his interrupt sequence to avoid the same possibilities.

6-57 Once the interrupt mode is established, the user may want to proceed to an unrelated operation, and return periodically to check for an interrupt (service request) by means of the bus status check. To leave the interrupt mode before a service request occurs, the user must send an "X" to the multiprogrammer system as illustrated in Example 10.

Example 11. Interrupt Mode of Operation, Individual Arming:

9830 Calculator

```
110 CMD "?U7","00240TATBTCT00460T"
                    .
                    .
                    .
120 IF STAT 13 > 1 THEN 500
130 GOSUB 2000
140 IF A4#64 THEN 500
150 CMD "?U7","00240TAX","?5W"
160 ENTER (13,*) A
170 IF A < 10000 THEN 200
180 X = A - 10000
190 CMD "?U7","00040TAT"
200 CMD "?U7","00240TBX","?5W"
210 ENTER (13,*) A
220 IF A < 10000 THEN 250
230 Y = A - 10000
240 CMD "?U7","00040TBT"
250 CMD "?U7","00240TCX","?5W"
260 ENTER (13,*) A
270 IF A < 10000 THEN 500
280 Z = A - 10000
290 CMD "?U7","00040TCT"
                    .
                    .
                    .
500 REM CONTINUE PROGRAM
```

9820/21 Calculators

```
CMD "?U7","00240TATBTCT00460T"
                    .
                    .
                    .
IF RDS 13 > 1; GTO "CONT"
GSB "SPOLL"
IF R4≠64; GTO "CONT"
CMD "?U7","00240TAX","?5W"
FMT *; RED 13, A
IF A ≤ 9999; JMP 3
A - 10000→X
CMD "?U7","00040TAT"
CMD "?U7","00240TBX","?5W"
FMT *; RED 13, A
IF A ≤ 9999; JMP 3
A - 10000→Y
CMD "?U7","00040TBT"
CMD "?U7","00240TCX","?5W"
FMT *; RED 13, A
IF A ≤ 9999; GTO "CONT"
A - 10000→Z
CMD "?U7","00040TCT"
                    .
                    .
                    .
"CONT" CONTINUE PROGRAM
```

Explanation:

110 Control word with ISL and SYE on is sent to MP. Cards in slots 401 ("A"), 402 ("B") and 403 ("C") are sequentially armed by address words "AT", "BT", and "CT" thus initiating the interrupt sequence for each card. Another control word ("00460T") establishes the interrupt mode by turning IEN, SYE, and TME on. (With TME, IEN on, the service request will not be set until the first card(s) interrupts.)

120 Bus status check; if service not requested, go to continue main program, line 500.

130 Service requested; go to serial poll routine for multiprogrammer.

140 Examine MP status byte in register; if not equal to 64, service not requested. Go to main program line 500.

150 Multiprogrammer service request (first card has interrupted). Control word is sent with ISL, SYE on, followed by an address word ("AX") to start IRQ poll. "AX" permits a read without gate sequence for first input card in slot 401.

160 ENTER statement allows return data from first card to be stored in variable "A".

170 Return data from first card is examined for IRQ. If IRQ is not present, first card has not interrupted; go to read without gate sequence for second card, line 200.

180 IRQ is present in return data word from first card (first card had caused interrupt and its' return data is valid). Subtract IRQ from data and store in "X".

190 Disarm first card. Control word with SYE on establishes output mode followed by data word "AT" to disarm (reset) the card.

200-240 Continue IRQ poll for second card in slot 402, by doing read without gate, IRQ check, and disarming procedure (if second card had interrupted).

250-290 Complete IRQ poll for third (last card in slot 403. IRQ procedures are the same for each card.

Table 6-2. Requirements for Cards with Interrupt Capability

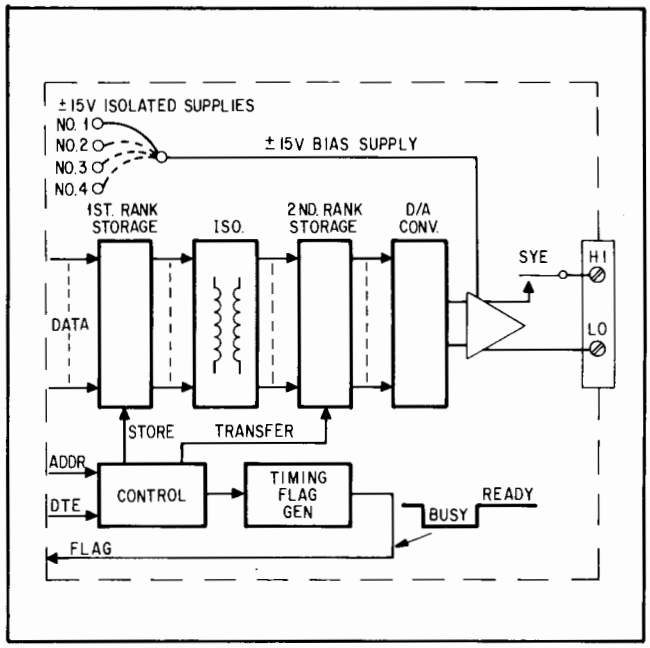
Card Model	Significance Of Data Sent In Output Mode	Arming Requirements (W6 Out)	Arming Requirements (W6 In)	Recycling Requirements	Disarming Requirements	Interrupting Requirements
69431A Digital Input		ISL on TME off Address true Gate	IEN on TME on Gate	ISL on TME off Address true Gate	ISL off TME off Address true Gate	External device returns trailing edge of flag
69436A Process Interrupt	All logical "1's" initializes card. Interrupting word recycles card. All logical "0's" disarms card.	ISL on TME off Address true Gate	IEN on TME on Gate	ISL off TME off Address true Data = Interrupting word. Gate	ISL off TME off Address true Data = logical "0's". Gate	Any one or more of twelve paired positive or negative edge detectors tripped.
69600A Timer	Binary representation of pulse duration. All logical "0's" disarms card.	ISL on TME off Address true Gate	IEN on TME on Gate	ISL off TME off Address true Data = Duration of next pulse. Gate	ISL off TME off Address true Data = logical "0's". Gate	End of programmed pulse
69434A Event Sense	Reference word	ISL on TME off Address true Gate	IEN on TME on Gate		ISL off TME off Address true Gate	External Word IS = Ref. word IS ≠ Ref. word IS > Ref. word IS < Ref. word

6-58 INDIVIDUAL CARD PROGRAMS

6-59 The following paragraphs provide programming examples for each type of plug-in card that can be presently used in a multiprogrammer system. The cards are presented in numerical sequence according to the card model number (69321 through 69601). In addition to the individual card programming examples, a brief functional description and a sample test program are provided for each type of card. Block diagrams and input/output connector diagrams (if applicable) are provided as programming aids. Any subroutines that may be used (e. g. decimal to octal conversion subroutine 2400) can be found in Appendix A.

6-60 D/A Voltage Converter Card, 69321B

6-61 This card is a twelve bit bipolar (two's complement) digital to analog converter which may be programmed to output a voltage in the range of -10.240 through +10.235 volts. An explanation of the relationship between an output voltage and the corresponding twelve bit binary code



69321B Block Diagram

is given in paragraph A-35 of Appendix A. The following discussion will familiarize the user with programming the D/A voltage converter card, 69321B. Unless otherwise specified, all examples will assume the D/A voltage card is installed in unit Ø, Slot 4Ø2 (B), and a voltage regulator card 69351A (which supplies the required ±15V bias voltages) is installed in unit Ø, slot 6ØØ.

6-62 Calculating Data Value. The following steps will yield the octal data value required to program the 69321B to a desired output voltage. Disregard the sign when making calculations. (If it is desired to write the program in decimal values,the decimal to octal subroutine must be used, see Appendix A, paragraph A-50).

1. Divide the desired output voltage in volts by .005 (the LSB).

2. For negative voltages only, subtract the absolute value yielded in step 1 from 4096.

3. Calculate the octal equivalent of the decimal value yielded in step 1 or 2 (see paragraph A-21 in Appendix A).

For example, to calculate the octal data value required for +5 volts;

5 ÷ .005 = 1000
octal equivalent of 1000 = 1750

or, to calculate the octal data value required for -5 volts;

5 ÷ .005 = 1000
4096-1000 = 3096
octal equivalent to 3096 = 6030

6-63 Programming Output Voltage Using Data Constants. Example 12 illustrates using a data constant of 6030_8 to program the output voltage of a 69321B card to -5 volts. A control word is sent containing SYE and DTE, followed by a data word containing the card slot address ("B") and the data constant. The "T" gate code is included with each word.

6-64 Programming Output Voltages Using Data Variables. In some cases it may be desirable to use a data variable to program the output voltage. The variable can be changed as a function of the user's program. In example 13, "A2" (or "R2" when using 9820/21A calculators) is the data variable which specifies the output voltage desired. The variable, of course, must be an octal value. If the program is written in decimal values, the decimal to octal conversion subroutine must be used.

6-65 Although a 69321B card returns a CTF flag as a function of internal timing circuits, TME is not included in the control words of the programming examples. This is because the output statement execution time of 9820/21 and 9830 calculators is greater than the period of the internal timing circuits on the card. Therefore, TME is not useful when programming this card. A description of the timing mode is given in paragraph 6-30.

6-66 Changing Output Voltages of Multiple 69321B Cards Simultaneously. The dual rank storage feature of 69321B cards and the DTE control mode allow the output voltages of any number of 69321B cards to be changed simultaneously. A detailed description of this technique is given in Chapter II (paragraphs 2-28 and 2-29) and in paragraph 6-27 of this Chapter.

6-67 In examples 12 and 13, DTE and SYE were programmed on in a control word prior to sending a data word to the card. In this case when the data word is received, it is loaded into the addressed card and immediately converted to an analog voltage that is present at the output of the card.

6-68 When it is desired to change the outputs of multiple 69321B cards simultaneously, a control word is sent with DTE off, followed by the appropriate address words, then another control word with DTE on. As each data word is received, the data will be loaded into the first rank storage of the addressed card. When a subsequent control word with DTE is received, all previously addressed D/A cards will simultaneously transfer the contents of first rank storage into second rank storage and the output D/A conversion circuits.

6-69 Example 14 illustrates using DTE with two 69321B

Example 12. Programming Output Voltage Using Data Constants:

9830A Calculator

```
11Ø CMD "?U7","ØØ14ØTB6Ø3ØT"
.
.
.
CONTINUE PROGRAM
```

9820/21A Calculators

```
CMD "?U7","ØØ14ØTB6Ø3ØT"
.
.
.
CONTINUE PROGRAM
```

Example 13. Programming Output Voltages Using Data Variables:

9830A Calculator

```
11Ø CMD "?U7", "ØØ14ØT"
12Ø FORMAT F1ØØ5.Ø
13Ø OUTPUT (13, 12Ø) "B"A2"T";
        .
        .
        .
CONTINUE PROGRAM
```

9820/21A Calculators

```
CMD "?U7", "ØØ14ØT"
FMT Z, "B",FXD*.Ø;WRT 13,R2
FMT Z, "T",FXD*.Ø;WRT 13
        .
        .
        .
CONTINUE PROGRAM
```

cards. In this case, the first rank storage of the cards located in multiprogrammer slots 4Ø2 (B) and 4Ø3 (C) is loaded with new data. After both cards have received new data, DTE is turned on by the control word, "ØØ14ØT", resulting in both cards transferring data from first rank to second rank storage and converting the data to analog outputs.

Example 14. Simultaneously Changing Output Voltages of Multiple 69321B Cards:

```
CMD "?U7", "ØØØ4ØTB175ØTC6Ø3ØTØØ14ØT"
```

6-70 Sample Test Program. The following sample program allows the user to program a 69321B card to any value within its capability. If an output is requested which exceeds the capability of the card, it will not be processed by the program.

6-71 Notice that this program could be used as a voltage programming subroutine by merely eliminating lines 1Ø and 2Ø of the 9830A program or the first line of the 9820/21A program, and changing the last statement from "END" to "RETURN". Of course, the line numbers of the 9830 program would have to be changed, and the first line of the 9820/21A program would require a label (e. g., "VOUT"). When used this way, the main program would supply the desired voltage (in volts) as variable "A" whenever the subroutine was called.

6-72 Test Procedure. Perform the following steps when using the 69321B sample test program.

1. Load the program into the calculator, being sure to include the appropriate subroutines.

2. Connect a voltmeter capable of measuring bipolar voltages between terminals A and B of the output terminal block on the 69321B card. The high terminal of the voltmeter should be connected to terminal A.

3. Depressing RUN and EXECUTE on the 9830A calculator or END, EXECUTE, and RUN PROGRAM on 9820/21A calculators will allow the program to run. The calculator will pause with "ENTER VOLTAGE" displayed. Entering the desired voltage, then depressing EXECUTE (or RUN PROGRAM for 9820/21A calculators) will cause the 69321B card to program the desired voltage.

6-73 Power Supply/Amplifier Control Cards, 69325A – 69328A

6-74 These resistance output cards are designed to control the gain, voltage, and current limit of Hewlett-Packard Bipolar Power Supply/Amplifiers (BPS/A).

1. Model 69325A controls the magnitude and polarity of the BPS/A output voltage. During the period that the resistance output of the card is changing, a hold command from the card causes the BPS/A voltage to remain constant,

69321B Sample Test Program:

9830A Calculator

```
1Ø DISP "ENTER VOLTAGE";
2Ø INPUT A
3Ø IF A<–1Ø.24 OR A>1Ø.235 THEN 9Ø
4Ø A=A/.ØØ5
5Ø GOSUB 24ØØ
6Ø CMD "?U7", "ØØ14ØT"
7Ø FORMAT F1ØØ5.Ø
8Ø OUTPUT (13, 7Ø) "B" A2"T;
9Ø END
```

9820/21A Calculators

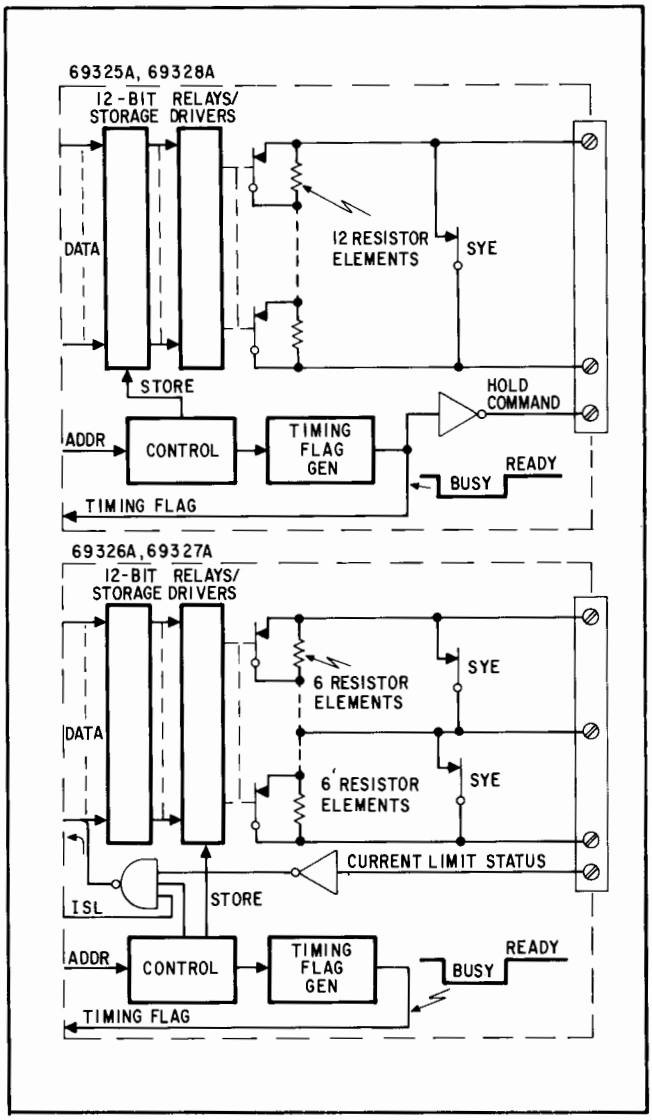
```
ENT "ENTER VOLTAGE", A
IF (A≤–1Ø.241) + (A>1Ø.235); JMP 6
A/.ØØ5→A
GSB "OCT"
CMD "?U7", "ØØ14ØT"
FMT Z, "B", FXD * .Ø; WRT 13, R2
FMT Z, "T", FXD * .Ø; WRT 13
END
```

and then make a smooth transition to the new output voltage.

2. The 69326A or 69327A card is used to independently control the positive and negative current limits of a BPS/A, and to monitor the status of the BPS/A current limit.

3. The 69328A is used to program the gain of a BPS/A from zero (no gain) to the full gain value of X2, X4, X8,X20, or X40, depending on the particular model of the HP BPS/A being controlled.

6-75 The following discussion will familiarize the user with programming the power supply/amplifier control cards, 69325A-69328A. All examples will assume the card is installed in unit Ø, slot 4Ø1, (A). For 69325A and 69328A cards, the resistance output terminals are A and B. For 69326A-69327A cards, the negative current limit control will be referred to as channel 1, while the positive current limit control will be referred to as channel 2. Channel 1 resistance output terminals are C and B, while channel 2 resistance output terminals are A and B.



69325A/28A and 69326A/27A Block Diagrams

6-76 Resistance Outputs. Octal data is transmitted to program one 12-bit resistance output from a BPS/A control card as follows:

1. Voltage Control Card 69325A: 12-bits in 2's complement form program the resistance output which controls the voltage magnitude and polarity of a BPS/A. (Refer to Paragraph A-30 in Appendix A for an explanation of 2's complement coding.)

2. Current Control Cards 69326A and 69327A: Each card provides two-identical 6-bit resistance channels. Channel 1 controls the negative current limit and channel 2 controls the positive current limit of a BPS/A. The two least significant octal digits in the data field program the negative current limit (channel 1). Two most significant digits program the positive current limit (channel 2).

3. Gain Control Card 69328A: 12-bits program the gain of a BPS/A.

6-77 Table 6-3 is provided as an aid in calculating data values (in decimal) necessary to program the control cards to specific output resistances. The decimal value obtained from using the applicable formula must then be converted to an octal value before being transmitted to the control card.

6-78 Assume that it is desired to program the output resistance of a 69325A card to 12,000 ohms. Using the applicable formula in Table 6-3, the data value is calculated as follows:

$$D = \frac{12{,}000}{5} - 2048$$

$$D = 2400 - 2048$$

$$D = 352$$

Octal equivalent of 352 = 540 (see Paragraph A-21 in Appendix A).

6-79 69325A Voltage Programming Formulas. Table 6-4 is provided as an aid in calculating the data value necessary to program the desired output voltage of BPS/A's. Make calculations as follows:

1. Using Table 6-4, select the formula associated with the applicable BPS/A model, output range, and polarity.

2. Insert the desired voltage output value as V in the formula. The desired output value must not exceed the

Table 6-3. 69325A – 69328A Resistance Formulas

BPS/A Control Card	Date Value (D) in Decimal Necessary To Program Desired Output Resistance (R) in Ohms
69325A	If desired resistance (R) is less than 10,240 ohms: D = (R ÷ 5) + 2048 If desired resistance (R) is from 10,240 to 20,475 ohms: D = (R ÷ 5) – 2048
69326A	Channel 1 or Channel 2: D = R ÷ 160
69327A	Channel 1 or Channel 2: D = R ÷ 80
69328A	D = R ÷ 5

maximum limits specified in the 69325A card manual.

3. Calculate the required decimal value.

4. Calculate the octal equivalent of decimal value obtained in step 3.

6-80 For example, to calculate the octal value required to program + 12.9 volts from a 6825A BPS/A:

1. D = V ÷ .01
2. D = 12.9 ÷ .01
3. D = 1290
4. Octal equivalent of 1290 = 2412

6-81 69326A/69327A Current Limit Formulas. Table 6-5 is provided as an aid in calculating the data value necessary to program the positive and negative current limits of BPS/A's. Make calculations as follows:

1. Using Table 6-5, select the formula associated with the applicable control card and BPS/A model.

2. Insert the desired negative current limit (channel 1) value as I in the selected formulas (use absolute value). The desired value must not exceed the maximum limits specified in the 69326A or 69327A card manual.

3. Calculate the decimal value required.

Table 6-4. 69325A Voltage Programming Formulas

BPS/A Models	Data Value (D) In Decimal Necessary To Program BPS/A To Desired Output Voltage (V) *			
	BPS/A High Range		BPS/A Low Range	
	Positive	Negative **	Positive	Negative **
6825A/6830A	D = V ÷ .01	D = 4096 – (V ÷ .01)	D = V ÷ .0025	D = 4096 – (V ÷ .0025)
6826A/6831A	D = V ÷ .025	D = 4096 – (V ÷ .025)	D = V ÷ .0025	D = 4096 – (V ÷ .0025)
6827A/6832A	D = V ÷ .05	D = 4096 – (V ÷ .05)	D = V ÷ .005	D = 4096 – (V ÷ .005)

* V is specified in volts

** Use absolute value of V, disregard sign.

Table 6-5. 69326A/69327A Current Limit Formulas

Current Control Card	BPS/A Models	Data Value (D) in Decimal Necessary To Program BPS/A To Desired Positive or Negative Current Limit (I) *
69326A	6825A/6830A	D = I ÷ 32
	6826A/6831A	D = I ÷ 16
69327A	6827A/6832A	D = I ÷ 8

* I is specified in mA. Use absolute value in making calculations, disregard sign.

Table 6-6. 69328A Gain Formulas

BPS/A Models	Data Value (D) in Decimal Necessary To Program BPS/A To Desired Gain (G)	
	High Range	Low Range
6825A/6830A	D = G ÷ (4 ÷ 2048)	D = G ÷ (1 ÷ 2048)
6826A/6831A	D = G ÷ (10 ÷ 2048)	D = G ÷ (1 ÷ 2048)
6827A/6832A	D = G ÷ (20 ÷ 2048)	D = G ÷ (2 ÷ 2048)

4. Repeat steps 2 and 3 for the desired positive current limit (channel 2).

5. Calculator the octal equivalent of values obtained in steps 3 and 4.

6. Multiply the octal value obtained for the positive current limit (channel 2) by 100 (decimal).

7. Add the octal values obtained for channel 1 (negative) and channel 2 (positive) together.

6-82 For example, to calculate the octal value required to program a 6826A BPS/A to −160mA (channel 1) and +320mA (channel 2) proceed as follows:

1. D = I ÷ 16
2. D = 160 ÷ 16
3. D = 10 (channel 1)
4. D = 320 ÷ 16
   D = 20 (channel 2)
5. channel 1 octal = 12
   channel 2 octal = 24
6. 24 X 100 = 2400
7. 12 + 2400 = 2412

6-83 69328A Gain Formulas. Table 6-6 is provided as an aid in calculating the data value necessary to program the desired gain of BPS/A's. Make calculations as follows:

1. Using Table 6-6, select the formula associated with the applicable BPS/A model and output range.

2. Insert the desired gain as G in the formula. The desired gain must not exceed the limits specified in the 69328A manual. Note that the maximum gain is 1 LSB less than the maximum specified in the 69328A manual.

3. Calculate the required decimal value.

4. Calculate the octal equivalent of value obtained in step 3.

6-84 Programming Examples. Examples 15 and 16 illustrate programming an output from a BPS/A control card. A control word is sent first containing SYE only, followed by a data word containing the card slot address, and the output data desired.

6-85 Although a BPS/A control card returns a CTF flag as a function of internal timing circuits, TME is not included in the control words of the programming examples. This is because the output statement execution time of 9820/21A and 9830A calculators is approximately the same as the period of the internal timing circuits on the card. If the users system requires each card to complete its operation before programming the next card, the user has the choice of programming one card per program statement, or using TME in the control words. If TME is used, the multiprogrammer system will set its service request when the card(s)

finish processing the output data. In this case, a serial poll of the multiprogrammer system must be done to reset the service request.

6-86 Example 15 illustrates using a data constant of 2412_8 to program a BPS/A control card located in slot 401(A).

6-87 Example 16 illustrates using a data variable ("A2" or "R2") to program a BPS/A control card.

6-88 Reading Status from 69326A/69327A Cards. The 69326A and 69327A cards examine the current limit status of the BPS/A. Example 17 illustrates the format required to read the status. In Example 17, the return data word is input to variable "A". The return data word will equal 1 if the BPS/A is in current limit or 0 if it is not. Note that the "X" (read without gate) code is used when reading the return data word. The "X" must be used to prevent changing the current limit values stored on the card.

6-89 69325A and 69328A Sample Test Program. This program will enable the user to verify operation of the 69325A and 69328A cards. The sample program allows the user to program a resistance value from a 69328A card. To program a 69325A card, line 40 of the 9830A program, or the third line of the 9820/21A program must be rewritten as follows:

9830A : 40A = (A/5) −2048
9820/21A : (A/5) −2048→A

NOTE

It is not required to take into account the equation for resistance less than 10,240 ohms (see Table 6-3) in this program. The decimal to octal subroutine (GOSUB 2400) is designed to accept negative numbers. Also, since resistance values must always be a multiple of the resolution (LSB) of the card, any value which is not a multiple of the card LSB is rounded off to the nearest value which is a multiple of the LSB.

6-90 69325A/69328A Test Procedure. Perform the following steps when using the 69325A/69328A sample test program.

1. Load the program into the calculator, being sure to include the appropriate subroutines.

2. Depressing RUN and EXECUTE on the 9830A cal-

Example 15. Programming a BPS/A Using Data Constants

9830A Calculator

```
110 CMD "?U7",'00040T"
110 CMD "?U7","00040TA2412T"
      .
      .
      .
CONTINUE PROGRAM
```

9820/21A Calculators

```
CMD "?U7","00040T"
CMD "?U7","00040TA2412T"
      .
      .
      .
CONTINUE PROGRAM
```

Example 16. Programming a BPS/A Using Data Variables:

9830A Calculator

```
120 FORMAT F1005.0
130 OUTPUT (13,120) "A"A2"T";
      .
      .
      .
CONTINUE PROGRAM
```

9820/21A Calculators

```
FMT Z, "A", FXD *.0; WRT 13, R2
FMT Z, "T", FXD *.0; WRT 13
      .
      .
      .
CONTINUE PROGRAM
```

Example 17. Reading BPS/A Current Limit Status

9830A Calculator

```
140 CMD "?U7","00240TAX","?5W"
150 ENTER (13,*) A
```

9820/21A Calculators

```
CMD "?U7","00240TAX","?5W"
FMT *; RED 13, A
```

culator, or END, EXECUTE, and RUN PROGRAM on 9820/21A calculators will allow the program to run.

3. The calculator will stop with "ENTER RESISTANCE" displayed. The user must enter a resistance value, no greater than the maximum capability of the card then depress EXECUTE on the 9830A calculator or RUN PROGRAM on 9820/21 calculators.

4. After running program, the calculator will display "TEST RESISTANCE". At this point, the user should use an ohmmeter to measure the resistance output between terminals A and B on the card.

6-91 69326A/69327A Sample Test Program. This program enables the user to verify operation of the 69326A and 69327A cards. The program allows the user to program separate resistance values from channels 1 and 2 of the 69326A card. To program 69327A cards, lines 40 and 90 of the 9830 program or the third and seventh line of the 9820/21 program must be changed so variable "A" is divided by the resolution (80) of the 69327A card as given in Table 6-3.

6-92 69326A/69327A Test Procedure. Perform the following steps when using the 69326A/69327A sample test program.

1. Load the program into the calculator, being sure to include the appropriate subroutines.

2. Depressing RUN and EXECUTE on the 9830 calculator, or END , EXECUTE, and RUN PROGRAM on 9820/21 calculators will allow the program to run.

3. The calculator will stop with "ENTER R1" displayed. The user must enter a resistance value for channel 1, no greater than the maximum capability of the card, then depress EXECUTE on the 9830 calculator or RUN PROGRAM on 9820/21 calculators.

4. The calculator will stop with "ENTER R2" displayed. The user must enter a resistance value for channel 2, no greater than the maximum capability of the card, then depress EXECUTE on the 9830 calculator or RUN PROGRAM on 9820/21 calculators.

5. After running the program, the calculator will display "TEST RESISTANCE". At this point, the user should use an ohmmeter to measure the resistance output as follows.

a. The resistance output of channel 1 may be measured between card terminals C and B.

b. The resistance output of channel 2 may be measured between card terminals A and B.

6-93 Relay Output Card, 69330A

6-94 This card provides 12, independent, SPST normally open (Form A) relays. Output data will be in the form of contact closures. Two additional relays are provided for external circuit gate/flag operation. An output connection is provided to route the 12 data outputs and the gate/flag signals to the external devices.

6-95 The following discussion will familiarize the user with programming the relay output card, 69330A. All examples will assume the relay output card is installed in unit 0, slot 404 (D).

69325A/69328A Sample Test Program

9830A Calculator

```
10 GOSUB 2000
20 DISP "ENTER RESISTANCE";
30 INPUT A
40 A = A/5
50 GOSUB 2400
60 CMD "?U7",",00040T"
70 FORMAT F1005.0
80 OUTPUT (13,70)"A"A2"T";
90 DISP "TEST RESISTANCE"
100 END
```

69326A/69327A Sample Test Program

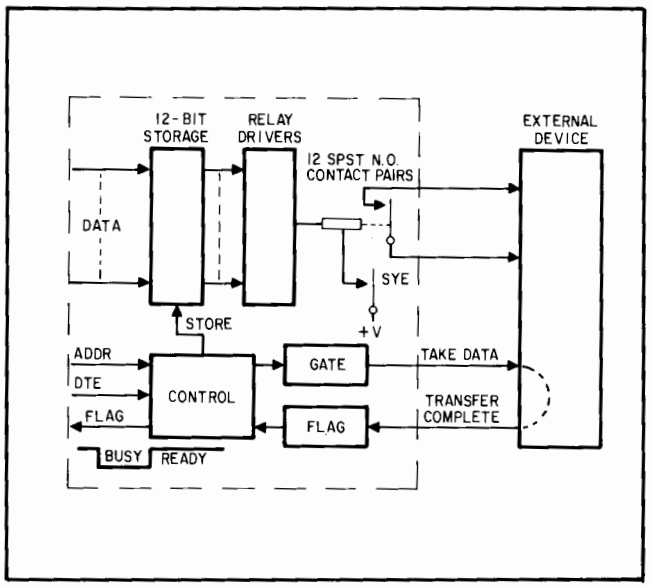
9830A Calculator

```
10 GOSUB 2000
20 DISP "ENTER R1";
30 INPUT A
40 A = A/160
50 GOSUB 2400
60 B = A2
70 DISP "ENTER R2";
80 INPUT A
90 A = A/160
100 GOSUB 2400
110 C = A2 * 100
120 A2 = B + C
130 CMD "?U7", "00040T"
140 FORMAT F1005.0
150 OUTPUT (13,140) "A"A2"T";
160 DISP "TEST RESISTANCE"
170 END
```

6-96 Operating Modes. Basically there are two modes of operation which apply to this card:

1. Automatic Handshake Mode (TME off): Used to output data to an external device when it is not necessary to handshake with the device.

2. Timing Mode (TME on): Used to output data to an external device, wait for an indication from the device that the data has been accepted, then cause the bus service request to be set.



69330A Block Diagram

NOTE

When using this card without an external gate/flag timing circuit, the card's gate output must be jumpered to its flag input (see Paragraph 6-37).

6-97 Automatic Handshake Mode. This mode requires sending a control word containing SYE and DTE only, followed by a data word containing the card slot address, and the output data desired. Example 18 illustrates using a data constant of 7777_8 to program the output of a 69330A card, located in slot 4Ø4 (D).

6-98 Example 19 illustrates using a data variable to program the output of a 69330A card. In this example, "A2" (or "R2" when using 9820/21A calculators) is the variable which specifies the desired output data.

6-99 Timing Mode. In this mode, a control word (with DTE and SYE on) followed by a data word, and then another control word (with DTE, SYE, and TME on) are sent to the multiprogrammer. The 69330A card will begin a timing sequence with its external device upon receipt of the "T" in its data word. When the timing sequence is completed, the multiprogrammer system will set its' service request (see Paragraphs 6-34 through 6-36).

Example 18. Programming Data Constants:

9830A Calculator

```
11Ø CMD "?U7", "ØØ14ØTD7777T"
      .
      .
      .
CONTINUE PROGRAM
```

9820/21A Calculators

```
CMD "?U7", "ØØ14ØTD7777T"
      .
      .
      .
CONTINUE PROGRAM
```

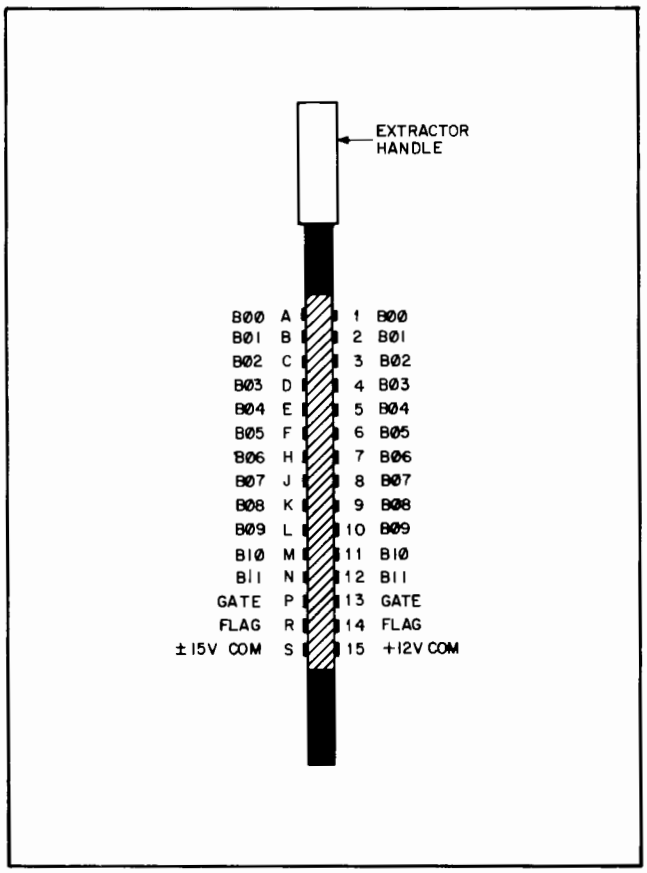
Example 19. Programming Data Variables:

9830A Calculator

```
11Ø CMD "?U7", "ØØ14ØT"
12Ø FORMAT F1ØØ5.Ø
13Ø OUTPUT (13,12Ø) "D"A2"T";
      .
      .
      .
CONTINUE PROGRAM
```

9820/21A Calculators

```
CMD "?U7", "ØØ14ØT"
FMT Z, "D", FXD * .Ø; WRT 13, R2
FMT Z, "T", FXD * .Ø; WRT 13
      .
      .
      .
CONTINUE PROGRAM
```



69330A Output Connector

NOTE

DTE and TME must be programmed on together because the 69330A card cannot drive the CTF line unless DTE is on (see Paragraph 6-37).

6-100 Example 20 is written in the format used to output data variables in the timing mode. In this example, "A2" (or "R2" when using 9820/21A calculators) is the variable data.

6-101 Example 20 can easily be changed to output constant data (fixed octal values) by eliminating the second and third program lines, and rewriting the first line as follows:

CMD "?U7","00140TD7777T00160T"

6-102 In the rewritten first line, 7777 is a fixed octal value representing the desired data. In either example (variable or constant data) when the data is completed, the multiprogrammer system will set its service request.

6-103 Using DTE to Simultaneously Gate Data Into External Devices. Some applications may require data to be loaded into several 69330A cards, then simultaneously gated into external devices. In this case, a control word may be sent with DTE off, followed by appropriate address words, then another control word with DTE on. As each data word is received, the data will be immediately transferred to the output of the addressed card. However, the gate from the output card will not be enabled until a control word containing DTE is received (see Paragraph 6-24 and 6-25).

6-104 Sample Test Program. The 69331A Sample Test Program on page 6-24 will enable the user to verify the operation of 69330A relay output cards. Use test procedure in paragraph 6-105.

Example 20. Programming Data Variables In The Timing Mode:

9830A Calculator

```
110 CMD "?U7","00140T"
120 FORMAT F1005.0
130 OUTPUT (13,120) "D"A2"T00160T";
                .
                .
                .
200 IF STAT 13>1 THEN 500
210 GOSUB 2000
220 IF A4#64 THEN 500
230 REM DATA TRANSFER IS COMPLETE
                .
                .
                .
500 REM CONTINUE PROGRAM
```

9820/21A Calculators

```
CMD "?U7","00140T"
FMT Z,"D",FXD *.0;WRT 13,R2
FMT Z,"T00160T",FXD *.0;WRT 13
                .
                .
                .
IF RDS 13>1;GTO "CONT"
GSB "SPOLL"
IF R4≠64;GTO "CONT"
  (DATA TRANSFER IS COMPLETE)
                .
                .
                .
"CONT" CONTINUE PROGRAM
```

6-105 Test Procedure. Perform the following steps when using the sample test program.

1. Load the program into the calculator, being sure to include the appropriate subroutines.

2. On the output connector of the 69330A card, temporarily install a jumper between pins 13 and 14, and a second jumper between pins P and 15.

3. Depressing RUN and EXECUTE on the 9830A calculator or END, EXECUTE, and RUN PROGRAM on 9820/21A calculators will allow the program to run.

4. The calculator will stop with "ENTER PIN NO." displayed. The user must enter a pin number from 1 to 12, then depress EXECUTE on the 9830A calculator or RUN PROGRAM on 9820/21A calculators.

5. The calculator will stop with "TEST BIT" displayed. At this point, the user should use an ohmmeter to check the contact resistance of the relays. Relay outputs can be measured between a numbered pin, 1 through 12, and a corresponding lettered pin (i. e., 1 and A, 2 and B, 12 and N). All relay contacts should be open except for the contacts corresponding to the pin number that was entered.

6. After the bit has been tested, remove the jumper from pin 13 to 14. Depressing CONT and EXECUTE on the 9830A or RUN PROGRAM on the 9820/21 calculators will cause the calculator to hang-up (provide a blank display).

7. Momentarily short pins 13 and 14 together. The calculator will display "TEST COMPLETE".

6-106 Digital Output Card, 69331A

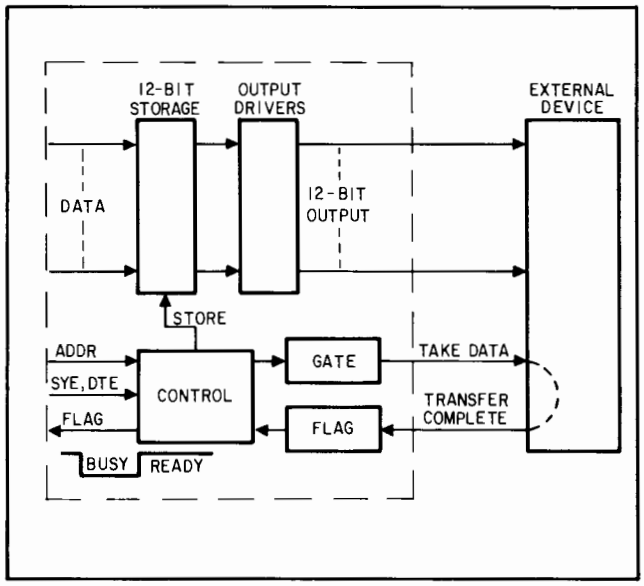
6-107 This card provides 12 bits with TTL/DTL-compatible logic levels as its output, and uses the gate/flag timing method of digital data transfer with an external device. An output connector is provided to route the 12 output bits and gate/flag signals to the external device.

6-108 The following discussion will familiarize the user, with programming the digital output card, 69331A. All examples will assume the output card is installed in unit Ø, slot 4Ø1 (A).

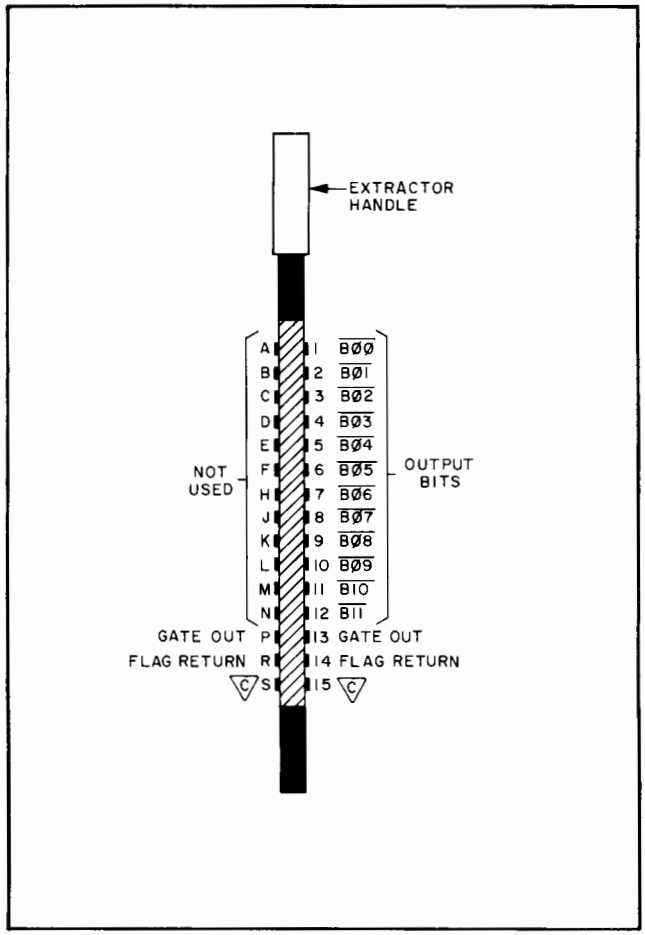
6-109 Operating Modes. Basically there are two modes of operation which apply to this card:

1. Automatic Handshake Mode (TME off): Used to output data to an external device when it is not necessary to handshake with the device.

2. Timing Mode (TME on): Used to output data to an external device, wait for an indication from the device that the data has been accepted, then cause the bus service request to be set.



69331A Block Diagram



69331A Output Connector

NOTE

When using this card without an external gate/ flag timing circuit, the card's gate output must be jumpered to its flag input (see Paragraph 6-37).

6-110 Automatic Handshake Mode. This mode requires sending a control word containing SYE and DTE only, followed by a data word containing the card slot address, and the output data desired. Example 21 illustrates using a data constant of 7777_8 to program the output of a 69331A card located in slot 401 (A).

6-111 Example 22 illustrates using a data variable to program the output of a 69331A card. In this example, "A2" (or "R2" when using 9820/21A calculators) is the variable which specifies the output data.

6-112 Timing Mode. In this mode, a control word (with DTE and SYE), followed by a data word, and then another control word (with DTE, SYE, and TME) are sent to the multiprogrammer. The 69331A card will begin a timing sequence with its external device upon receipt of the "T" in its data word. When the timing sequence is completed, the multiprogrammer system will set its service request (see Paragraphs 6-34 through 6-36).

NOTE

DTE and TME must be programmed on together because the 69331A card cannot drive the CTF line unless DTE is on (see Paragraph 6-37).

6-113 Example 23 is written in the format used to output data variables in the timing mode. In this example, "A2" (or "R2" when using 9820/21A calculators) is the variable data.

6-114 Example 23 can easily be changed to output constant data (fixed octal values) by eliminating the second and third program lines, and rewriting the first line as follows:

```
CMD "?U7", "00140TA7777T00160T"
```

6-115 In the rewritten first line, 7777 is a fixed octal value representing the desired data. In either example (variable or constant data), when the data transfer is complete the multiprogrammer system will set its service request.

6-116 Using DTE to Simultaneously Gate Data Into External Devices. Some card applications may require data to be loaded into several 69331A cards, then simultaneously gated into external devices. In this case a control word may be sent with DTE off, followed by appropriate address words, then another control word with DTE on. As each data word is received, the data will be immediately transferred to the output of the addressed card. However, the gate from the output card will not be enabled until a control word containing DTE is received (see Paragraphs 6-24 and 6-25).

6-117 Sample Test Program. The sample test program will enable the user to verify the operation of 69331A digital output cards.

6-118 Test Procedure. Perform the following steps when

Example 21. Programming Data Constants:

9830A Calculator

```
110 CMD "?U7", "00140TA7777T"
      .
      .
      .
CONTINUE PROGRAM
```

9820/21A Calculators

```
CMD "?U7", "00140TA7777T"
      .
      .
      .
CONTINUE PROGRAM
```

Example 22. Programming Data Variables

9830A Calculator

```
110 CMD "?U7", "00140T"
120 FORMAT F1005.0
130 OUTPUT (13,120) "A"A2"T";
      .
      .
      .
CONTINUE PROGRAM
```

9820/21A Calculators

```
CMD "?U7", "00140T"
FMT Z, "A", FXD * .0; WRT 13, R2
FMT Z, "T", FXD * .0; WRT 13
      .
      .
      .
CONTINUE PROGRAM
```

Example 23. Programming Data Variables In The Timing Mode:

9830A Calculator

```
11Ø CMD "?U7", "0Ø14ØT"
12Ø FORMAT F1ØØ5.Ø
13Ø OUTPUT (13,12Ø) "A"A2"T0Ø16ØT";
                .
                .
                .
2ØØ IF STAT 13 > 1 THEN 5ØØ
21Ø GOSUB 2ØØØ
22Ø IF A4 # 64 THEN 5ØØ
23Ø REM DATA TRANSFER IS COMPLETE
                .
                .
5ØØ REM CONTINUE PROGRAM
```

9820/21A Calculators

```
CMD "?U7", "0Ø14ØT"
FMT Z, "A", FXD * .Ø; WRT 13, R2
FMT Z, "T0Ø16ØT", FXD * .Ø; WRT 13
                .
                .
                .
IF RDS 13 > 1; GTO "CONT"
GSB "SPOLL"
IF R4 ≠ 64; GTO "CONT"
   (DATA TRANSFER IS COMPLETE)
                .
                .
"CONT" CONTINUE PROGRAM
```

69331A Sample Test Program

9830A Calculator

```
1Ø GOSUB 2ØØØ
2Ø DISP "ENTER PIN NO. ";
3Ø INPUT A
4Ø A = 2 ↑ (A-1)
5Ø GOSUB 24ØØ
6Ø CMD "?U7", "0Ø14ØT"
7Ø FORMAT F1ØØ5.Ø
8Ø OUTPUT (13,7Ø) "A"A2"T";
9Ø DISP "TEST BIT"
1ØØ STOP
11Ø CMD "?U7", "0Ø16ØTAT0ØØ4ØT"
12Ø DISP "TEST COMPLETE"
13Ø END
```

9820/21A Calculators

```
GSB "SPOLL"
ENT "ENTER PIN NO. ", A
2 ↑ (A-1)→A
GSB "OCT"
CMD "?U7", "0Ø14ØT"
FMT Z, "A", FXD * .Ø; WRT 13, R2
FMT Z, "T", FXD * .Ø; WRT 13
DSP "TEST BIT"
STP
CMD "?U7", "0Ø16ØTAT0ØØ4ØT"
DSP "TEST COMPLETE"
END
```

using the 69331A test program.

1. Load the program into the calculator, being sure to include the appropriate subroutines.

2. On the output connector of the digital output card, temporarily install a jumper between pins 13 and 14.

3. Depressing RUN and EXECUTE on the 9830 calculator, or END, EXECUTE, and RUN PROGRAM on 9820/21A calculators will allow the program to run.

4. The calculator will stop with "ENTER PIN NO." displayed. The user must enter a pin number from 1 to 12, then depress EXECUTE on the 9830 calculator or RUN PROGRAM on 9820/21A calculators.

5. The calculator will stop with "TEST BIT" displayed. At this point the user should use a voltmeter to measure the output voltage on pins 1 through 12. (All voltage readings are referenced to pin 15, (common).

a. On the standard digital output card, all pins shall read a positive voltage (+5 or +12), except for the pin that the user has previously entered into the program, which shall read approximately 0 volts.

b. On Option 073 output cards, all pins shall read approximately 0 volts, except for the pin that the user has previously entered into the program, which shall read a positive voltage (+5 or +12).

6. After the bit has been tested, remove the jumper from pin 13 to 14, depressing CONT and EXECUTE on the 9830A or RUN PROGRAM on the 9820/21A calculators will cause the calculator to hang-up (provide a blank display).

7. Momentarily short pins 13 and 14 together. The calculator will display "TEST COMPLETE".

6-119 Open Collector Output Card, 69332A

6-120 This digital output card is designed to drive lamps and relay coils utilizing an external dc power source. Twelve separately programmable outputs are available, each of which is open collector. The following discussion will familiarize the user with programming the open collector output card, 69332A. All examples will assume the output card is installed in unit Ø slot 4Ø5 (E).

6-121 Programming Examples. Examples 24 and 25 illustrate programming data from an open collector output card. A control word is sent first, containing SYE only, followed by a data word containing the card slot address, and the output data desired. Although SYE has no effect on this card, it is included in the control word to avoid disabling the outputs of previously programmed output cards.

6-122 Example 24 illustrates using a data constant of 7777_8 to program the output of a 69332A card located in slot 4Ø5 (E).

6-123 Example 25 illustrates using a data variable to program the output of a 69332A card. In this example "A2" (or R2" when using 9820/21A calculators) is the variable which specifies the output data desired. It should be kept in mind that this must be an octal value. If desired, the decimal to octal subroutine provided in Appendix A may be used to enable programs to be written with decimal values (see Paragraph 6-15).

6-124 Sample Test Program. This program will enable the user to verify the operation of 69332A open collector output cards. Since the data outputs from the card are open collector, it will be necessary for the user to provide a pull-up resistor for this test. Although the resistance value is not critical, the recommended resistance value is between 1K and 10K ohms.

6-125 Test Procedure. Perform the following steps when using the 69332A sample test program.

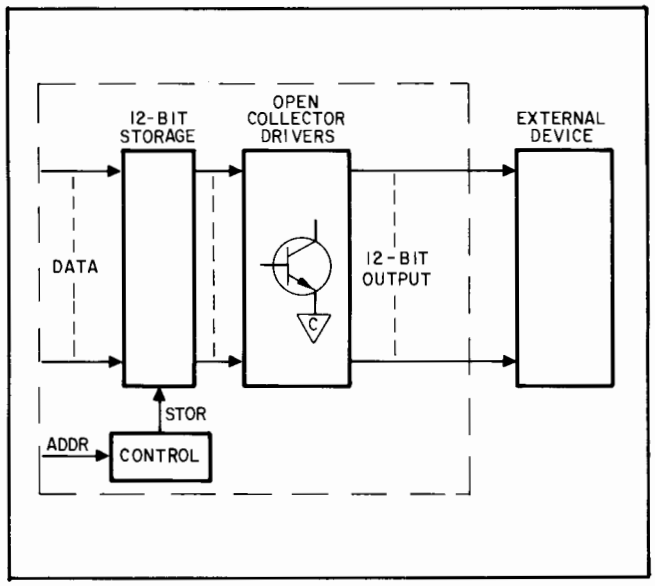
1. Load the program into the calculator, being sure to include the appropriate subroutines.

2. On the output connector of the open collector output card, temporarily install a pull-up resistor between pin 14 and any data output pin, 1 through 12, as desired.

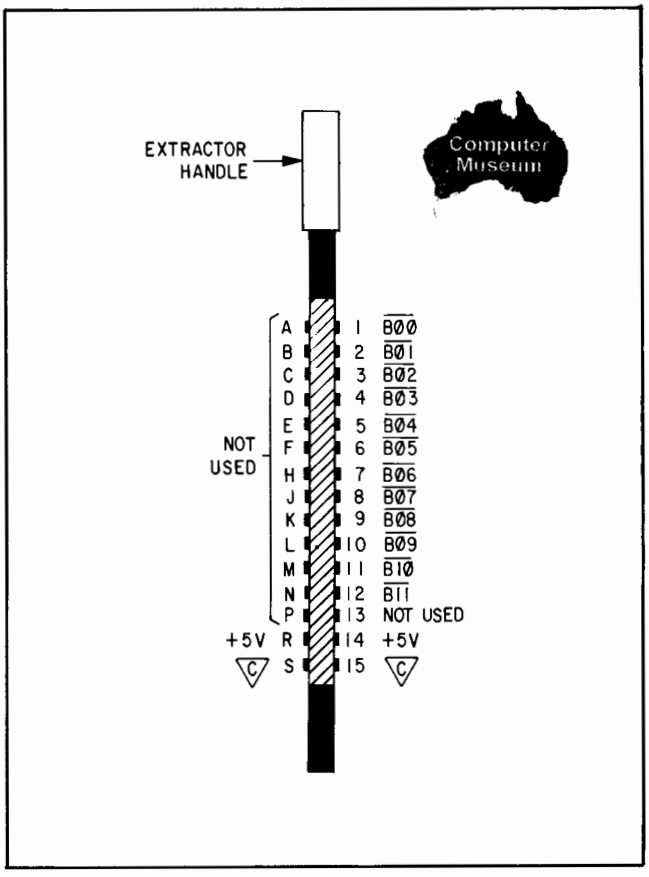
3. Depressing RUN and EXECUTE on the 9830A calculator or END, EXECUTE, and RUN PROGRAM on 9820/21A calculators will allow the program to run.

4. The calculator will stop with ENTER PIN NO. displayed. The user must enter the pin number that had the pull-up resistor installed in step 2, then depress EXECUTE on the 9830A calculator or RUN PROGRAM on 9820/21A calculators.

5. The calculator will stop with TEST BIT displayed. At this point the user should use a voltmeter to measure the



69332A Block Diagram



69332A Output Connector

Example 24. Programming Data Constants:

9830A Calculator

```
110 CMD "?U7", "00040TE7777T"
                .
                .
                .
CONTINUE PROGRAM
```

9820/21A Calculators

```
CMD "?U7", "00040TE7777T"
                .
                .
                .
CONTINUE PROGRAM
```

Example 25. Programming Data Variables:

9830A Calculator

```
110 CMD "?U7", "00040T"
120 FORMAT F1005.0
130 OUTPUT (13,120) "E"A2"T";
                .
                .
CONTINUE PROGRAM
```

9820/21A Calculators

```
CMD "?U7", "00040T"
FMT Z, "E", FXD * .0; WRT 13, R2
FMT Z, "T", FXD * .0; WRT 13
                .
                .
CONTINUE PROGRAM
```

output voltage between the pin number selected in step 2 and pin 15 (common).

a. On the standard open collector output card, the voltmeter will indicate approximately zero volts.

b. On Option 090 output cards, the voltmeter will indicate +5 volts.

6. Depress CONT and EXECUTE on the 9830A or RUN PROGRAM on the 9820/21A calculators. The calculator will display TEST COMPLETE, and the voltages measured in step 5 will be reversed. (i. e., a standard card will now indicate +5 volts, while an Option 090 card will indicate approximately zero volts.

6-126 Stepping Motor Control Card, 69335A

6-127 This card can be programmed to generate from 1 to 2047 squarewave pulses at either of two output terminals on the card. When applied to a stepping motor translator, these pulses are converted to clockwise and counterclockwise drive pulses for an associated stepping motor. The squarewave outputs of the 69335A can also be used for pulse-train update of supervisory control stations. The following discussion will familiarize the user with programming the stepping motor control card, 69335A. All examples will assume the stepping motor card is installed in unit 0, slot 414 (N).

69332A Sample Test Program

9830A Calculator

```
10 GOSUB 2000
20 DISP "ENTER PIN NO. ";
30 INPUT A
40 A = 2 ↑ (A-1)
50 GOSUB 2400
60 CMD "?U7", "00040T"
70 FORMAT F1005.0
80 OUTPUT (13,70) "E"A2"T";
90 DISP "TEST BIT"
100 STOP
110 CMD "?U7", "ET"
120 DISP "TEST COMPLETE"
130 END
```

9820/21A Calculators

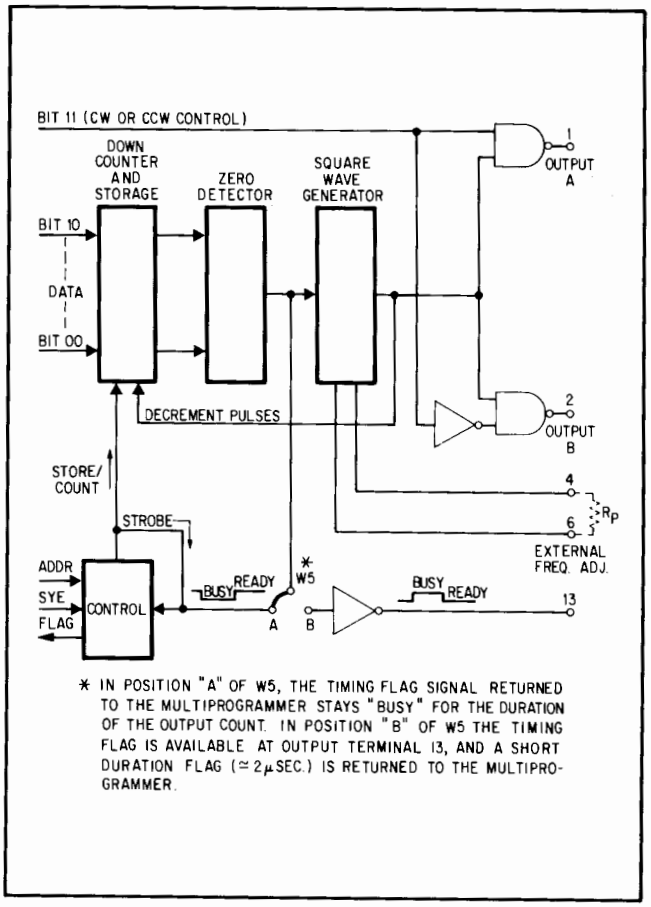
```
GSB "SPOLL"
ENT "ENTER PIN NO.", A
2 ↑ (A-1)→A
GSB "OCT"
CMD "?U7", "00040T"
FMT Z, "E", FXD * .0; WRT 13, R2
FMT Z, "T", FXD * .0; WRT 13
DSP "TEST BIT"
STP
CMD "?U7", "ET"
DSP "TEST COMPLETE"
END
```

6-128 Selecting Output Terminal and Number of Output Pulses. Data transmitted to this card is a combination of the number of output pulses desired and a user selected output terminal. For values up to octal 3777 (decimal 2047) the output will be from pin 1 of the output connector. If it is desired to obtain the output pulse train from pin 2, an octal 4000 must be added to the number of pulses desired (in octal). For example, 7777_8 would program 2047 pulses from pin 2.

6-129 Operating Modes. Basically there are two modes of operation which apply to this card.

1. Automatic Handshake Mode (TME off): Allows the user to program an output pulse train, consisting of a specified number of pulses, from either of two output circuits. Programmable selection of the appropriate output circuit allows the user to determine the direction of rotation of a stepping motor. This mode does not cause the multiprogrammer system to set its service request upon completion of the pulse train.

2. Timing Mode (TME on): Functions the same as automatic handshake mode, except that it causes the multiprogrammer system to set its service request upon completion of the pulse train.



69335A Block Diagram

6-130 Automatic Handshake Mode. This mode requires sending a control word with SYE only, followed by a data word containing the card slot address and output data (number of pulses and output terminal) desired.

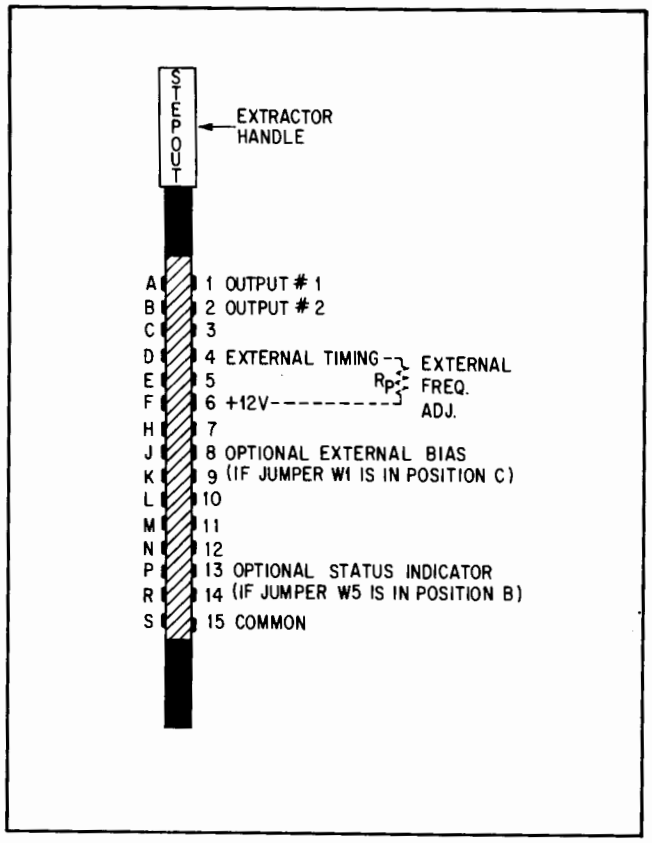
6-131 Example 26 illustrates using a data constant of 3777_8 to program 2047 output pulses from pin 1 of the 69335A card located in slot 414 (N).

6-132 Example 27 illustrates using a data variable to program the pulse train output. In this example, "A2" (or "R2" when using 9820/21A calculators) is the variable which specifies the output terminal and number of pulses desired. Again, it must be kept in mind that this must be an octal value (see Paragraph 6-15).

6-133 Timing Mode. In this mode, TME is programmed on in a control word after the pulse train has been initiated. With TME on, the bus service request will be set when the pulse train is completed.

6-134 Example 28 is written in the format used to program the output pulse train using data variables. In this example, "A2" (or "R2") is the variable data.

6-135 Example 28 can easily be changed to program the output pulse train using data constants by eliminating the



69335A Output Connector

Example 26. Programming Output Pulse Train Using Data Constants:

9830A Calculator

```
110 CMD "?U7", "00040TN3777T"
      .
      .
      .
CONTINUE PROGRAM
```

9820/21A Calculators

```
CMD "?U7", "00040TN3777T"
      .
      .
      .
CONTINUE PROGRAM
```

Example 27. Programming Output Pulse Train Using Data Variables

9830A Calculator

```
110 CMD "?U7", "00040T"
120 FORMAT F1005.0
130 OUTPUT (13,120) "N"A2"T";
```

9820/21A Calculators

```
CMD "?U7", "00040T"
FMT Z, "N", FXD * .0; WRT 13, R2
FMT Z, "T", FXD * .0; WRT 13
```

Example 28. Programming Output Pulse Train Using Data Variables in Timing Mode:

9830A Calculator

```
110 CMD "?U7", "00040T"
120 FORMAT F1005.0
130 OUTPUT (13,120)"N"A2"T00160T";
                .
                .
                .
200 IF STAT 13 > 1 THEN 500
210 GOSUB 2000
220 IF A4 # 64 THEN 500
230 REM PULSE TRAIN IS COMPLETE
                .
                .
                .
500 REM CONTINUE PROGRAM
```

9820/21A Calculators

```
CMD "?U7", "00040T"
FMT Z, "N", FXD * .0; WRT 13, R2
FMT Z, "T00160T", FXD * .0; WRT 13
                .
                .
                .
IF RDS 13 > 1; GTO "CONT"
GSB "SPOLL"
IF R4 ≠ 64; GTO "CONT"
   (PULSE TRAIN IS COMPLETE)
                .
                .
                .
"CONT" CONTINUE PROGRAM
```

second and third program lines, and rewriting the first line as follows:

```
CMD "?U7", "00040TN7777T00160T"
```

6-136 In the rewritten first line, 7777 is a fixed octal value representing the output terminal (pin 2) and the number (2047) of output pulses desired. In either example (variable or constant data), when the pulse train terminates, the multiprogrammer service request will be set.

6-137 Sample Test Program. The sample test program will enable the user to verify the operation of 69335A stepping motor control cards. For this example it will be assumed that an output pulse train from pin 1 will cause clockwise rotation of a stepping motor, while an output pulse train from pin 2 will cause counterclockwise rotation of a stepping motor. It will be up to the user to provide a means of monitoring the output pulse train.

6-138 Notice that lines 40 through 90 of the 9830A program, and the third through the eighth program line of the 9820/21A program, could be incorporated as a subroutine in a users program to program direction and magnitude of stepping motor rotation. When used this way, variable "A" must be supplied to the subroutine as a positive value for clockwise rotation, or a negative value for counterclockwise rotation. Of course, the line numbers must be changed for 9830A programs, a label must be used for 9820/21A programs (e. g., "STEP"), and "RETURN" must be added as the final line of the subroutine. Since

this example uses timing mode, the multiprogrammer system will set its service request whenever a pulse train terminates.

6-139 Test Procedure. Perform the following steps when using the 69335A test program.

1. Load the program into the calculator, being sure to include the appropriate subroutines.

2. Depressing RUN and EXECUTE on a 9830A calculator or END, EXECUTE and RUN PROGRAM on 9820/21A calculators will allow the program to run. The calculator will stop with "ENTER PULSES" displayed. The user must then enter the desired number of pulses (−2047 to +2047).

3. After entering the desired number of pulses, depressing EXECUTE on a 9830A calculator or RUN PROGRAM on 9820/21A calculators will allow the program to continue. The stepping motor card will transmit a pulse train approximately 12 volts in amplitude.

a. If a positive value had been entered in step 2, the output pulse train may be measured (or observed) between pins 1 and 15 (common).

b. If a negative value had been entered in step 2, the output pulse train may be measured (or observed) between pins 2 and 15 (common).

4. The calculator display will be blank while the pulse train is being output. This is caused by a status loop which will continually check for a multiprogrammer system service request, which will occur upon completion of the pulse train. Upon completion of the pulse train, the calculator will display "TEST COMPLETE". As shipped from the factory the output pulse rate is 100Hz. Therefore, the calculator display will remain blank for approximately 1 second for every 100 pulses entered in step 1.

6-140 D/A Current Converter Card, 69370A

6-141 This card provides a high speed, constant current output that is the analog equivalent of the digital input data. The output current range is from 0 to 20.475mA with a minimum programmable step change of 5μA. Current output levels are programmed in straight 12-bit binary form. The D/A voltage converter card block diagram on Page 6-12 also applies to the 69370A card.

6-142 The following discussion will familiarize the user with programming the D/A current converter card, 69370A. Unless otherwise specified, all examples will assume the D/A current card is installed in unit Ø, slot 4Ø2 (B), and a voltage regulator card 69351A (which supplies the required ±15V bias voltages) is installed in unit Ø, slot 6ØØ.

6-143 Calculating Data Value. The following steps will yield the octal data value required to program the 69370A to the desired output current. (Note that if the program is written in decimal values, the decimal to octal subroutine of Appendix A must be used in the program).

1. Divide the desired output current in milliamps by .005 (the LSB).

2. Calculate the octal equivalent of the value yielded in step 1 (see Paragraph A-21 in Appendix A).

69335A Sample Test Program

9830A Calculator

```
1Ø GOSUB 2ØØØ
2Ø DISP "ENTER PULSES";
3Ø INPUT A
4Ø IF A>−.ØØ1 THEN 6Ø
5Ø A=2Ø48+ABS (A)
6Ø GOSUB 24ØØ
7Ø CMD "?U7", "ØØØ4ØT"
8Ø FORMAT F1ØØ5.Ø
9Ø OUTPUT (13,8Ø) "N"A2"T1ØØ16ØT";
1ØØ B=STAT 13
11Ø IF B>1 THEN 1ØØ
12Ø GOSUB 2ØØØ
13Ø IF A4#64 THEN 1ØØ
14Ø DISP "TEST COMPLETE"
15Ø END
```

9820/21A Calculators

```
GSB "SPOLL"
ENT "ENTER PULSES", A
IF A≥−.ØØ1;JMP 2
2Ø48+ABS A→A
GSB "OCT"
CMD "?U7", "ØØØ4ØT"
FMT Z, "N", FXD*.Ø; WRT 13, R2
FMT Z, "T1ØØ16ØT", FXD*.Ø; WRT 13
RDS 13→B
IF B>1; JMP −1
GSB "SPOLL"
IF R4≠64; JMP −3
DSP "TEST COMPLETE"
END
```

For example, to calculate the octal data value required for 5 milliamps;

$5 \div .005 = 1000$

octal equivalent of 1000 = 1750

or, to calculate the octal data value required for 15 milliamps;

$15 \div .005 = 3000$

octal equivalent of 3000 = 5670

6-144 Programming Output Current Using Data Constants. Example 29 illustrates using a data constant of 1750_8 to program the output current of a 69370A card to 5mA. A control word is sent containing SYE and DTE, followed by a data word containing the card slot address ("B") and the data constant. Note that the gate code "T" is included with each word.

6-145 Programming Output Current Using Data Variables. In some cases it may be desirable to use a data variable to program the output current. The variable can be changed as a function of the user's program. In Example 30, "A2" (or "R2" when using 9820/21A calculators) is the data variable which specifies the output current desired.

6-146 Although a 69370A card returns a CTF flag as a function of internal timing circuits, TME is not included in the control words of the programming examples. This is because the output statement execution time of 9820/21 and 9830 calculators is greater than the period of the internal timing circuits on the card. Therefore, TME is not useful when programming this card, however, no harm is done if it is used. A description of the timing mode is given in Paragraph 6-30.

6-147 Changing Output Currents of Multiple 69370A Cards Simultaneously. The dual rank storage feature of 69370A cards and the DTE control mode allow the output currents of any number of 69370A cards to be changed simultaneously. A detailed description of this technique is given in Chapter II (Paragraphs 2-28 and 2-29) and in Paragraph 6-27 of this chapter.

6-148 In the previous example, DTE and SYE were programmed on in a control word prior to sending a data word to the card. In this case when the data word is received, it is loaded into the addressed card and immediately converted to an analog current that is present at the output of the card.

6-149 When it is desired to change the outputs of multiple 69370A cards simultaneously, a control word is sent with DTE off, followed by the appropriate address words, then another control word with DTE on. As each data word is received, the data will be loaded into the first rank storage of the addressed card. When a subsequent control word with DTE is received, all previously addressed D/A cards will simultaneously transfer the contents of first rank storage into second rank storage and the output D/A conversion circuits.

6-150 Example 31 illustrates using DTE with two 69370A cards. In this case, the first rank storage of the cards located in multiprogrammer slots 402 (B) and 403 (C) is loaded with new data. After both cards have received new data, DTE

Example 29. Programming Output Current Using a Data Constant:

9830A Calculator

```
110 CMD "?U7", "00140TB1750T"
      .
      .
      .
CONTINUE PROGRAM
```

9820/21A Calculators

```
CMD "?U7", "00140TB1750T"
      .
      .
      .
CONTINUE PROGRAM
```

Example 30. Programming Output Current Using Data Variables:

9830A Calculator

```
110 CMD "?U7", "00140T"
120 FORMAT F1005.0
130 OUTPUT (13,120) "B"A2"T";
      .
      .
      .
CONTINUE PROGRAM
```

9820/21A Calculators

```
CMD "?U7", "00140T"
FMT Z, "B", FXD * .0; WRT 13, R2
FMT Z, "T", FXD * .0; WRT 13
      .
      .
      .
CONTINUE PROGRAM
```

is turned on by the control word, "0Ø14ØT", resulting in both cards transferring data from first rank to second rank storage and converting the data to analog outputs.

Example 31. Simultaneously Changing Output Currents of Multiple 69370A Cards:

```
CMD "?U7", "0ØØ4ØTB175ØTC567ØTØØ14ØT"
```

6-151 **Sample Test Program.** The sample test program will allow the user to program a 69370A D/A card to any value within its capability. If an output is requested which exceeds the capability of the card, it will not be processed by the program. A load resistor will be required to ascertain the desired output current being programmed. The value of the resistor may be determined by the user, up to a maximum of 500 ohms. The accuracy of the resistor must be taken into account when determining the actual output current. For an absolute accuracy check, refer to the 69370A card manual.

6-152 Notice that this program could be used as a current programming subroutine by merely eliminating lines 1Ø and 2Ø of the 9830 program or the first line of the 9820/21 program, and changing the last statement from "END" to "RETURN". Of course, the line numbers of the 9830 program would have to be changed, and the first line of the 9820/21 program would require a label ("IOUT"). When used this way, the main program would supply the desired current (in milliamps) as variable "A" whenever the subroutine was called.

6-153 **Test Procedure.** Perform the following steps when using the 69370A sample test program.

1. Connect a load resistor between terminals A and B of the output terminal block on the 69370A card. Connect a voltmeter across the load resistor with the high input of the voltmeter at the end of the resistor connected to terminal A.

2. Load the program into the calculator, being sure to include the appropriate subroutines.

3. Depressing RUN and EXECUTE on the 9830A calculator or END, EXECUTE, and RUN PROGRAM on 9820/21A calculators will allow the program to run. The calculator will pause with "ENTER CURRENT" displayed. Entering the desired current, in milliamps, then depressing EXECUTE (or RUN PROGRAM for 9820/21A calculators) will cause the current D/A card to program the desired current.

4. The output current may then be determined by dividing the voltage measured across the load resistor by the value of the load resistor. Notice that unless the resistor accuracy and the D/A card accuracy are included in the calculations this should only be considered an approximate value. It will, however, be close enough to the programmed current to ascertain the card is working properly.

6-154 Breadboard Output Card, 69380A

6-155 The 69380A breadboard output card is a simple multiprogrammer interface card, which allows the user to design, build, and control special output circuits through the multiprogrammer system. The following discussion will familiarize the user with programming the breadboard output card, 69380A. All examples will assume the output card is installed in unit Ø, slot 4Ø6 (F).

6-156 **Programming Examples.** Examples 32 and 33 illustrate programming data from a breadboard output card. A control word is sent first, containing SYE only, followed by a data word containing the card slot address, and the output data desired. Although SYE has no effect on this card, as received from the factory, it is included in the control word to avoid disabling the outputs of previously programmed output cards.

6-157 Example 32 illustrates using a data constant of 7777_8 to program a 69380A card located in slot 4Ø6 (F).

6-158 Example 33 illustrates using data variable "A2" (9830A) or "R2" (9820/21A) to program the 69380A card.

69370A Sample Test Program

9830A Calculator

```
1Ø DISP "ENTER CURRENT";
2Ø INPUT A
3Ø IF A<Ø OR A>2Ø.475 THEN 9Ø
4Ø A = A/.ØØ5
5Ø GOSUB 24ØØ
6Ø CMD "?U7", "0Ø14ØT"
7Ø FORMAT F1ØØ5.Ø
8Ø OUTPUT (13,7Ø) "B"A2"T";
9Ø END
```

9820/21A Calculators

```
ENT "ENTER CURRENT", A
IF (A≤–.ØØ1) + (A≥2Ø.475); JMP 6
A/.ØØ5 – A
GSB "OCT"
CMD "?U7", "0Ø14ØT"
FMT Z, "B", FXD * .Ø; WRT 13, R2
FMT Z, "T", FXD * .Ø; WRT 13
END
```

6-159 Sample Test Program. The following program will enable the user to verify the operation of 69380A breadboard output cards.

6-160 Test Procedure. Perform the following steps when using the 69380A sample test program.

1. Load the program into the calculator, being sure to include the appropriate subroutines.

2. Depressing RUN and EXECUTE on the 9830A calculator or END, EXECUTE, and RUN PROGRAM on 9820/21A calculators will allow the program to run.

3. The calculator will stop with "ENTER PIN NO." displayed. The user must enter a pin number from 1 to 12, then depress EXECUTE on the 9830A calculator or RUN PROGRAM on 9820/21A calculators.

4. The calculator will stop with "TEST BIT" displayed. At this point the user should use a voltmeter to measure the output voltage on pins 1 through 12. (All voltage readings are referenced to pin 15 (common). All pins shall read approximately zero volts, except for the pin that the user has previously entered into the program, which shall read greater than 2.4 volts.

5. Repeating steps 2, 3, and 4 will enable the user to check each data output line on the card.

Example 32. Programming Data Constants:

9830A Calculator

```
11Ø CMD "?U7", "0ØØ4ØTF7777T"
      .
      .
CONTINUE PROGRAM
```

9820/21A Calculators

```
CMD "?U7", "0ØØ4ØTF7777T"
      .
      .
CONTINUE PROGRAM
```

Example 33. Programming Data Variables:

9830A Calculator

```
11Ø CMD "?U7", "0ØØ4ØT"
12Ø FORMAT F1ØØ5.Ø
13Ø OUTPUT (13,12Ø) "F"A2"T";
      .
      .
      .
CONTINUE PROGRAM
```

9820/21A Calculators

```
CMD "?U7", "0ØØ4ØT"
FMT Z, "F", FXD * .Ø; WRT 13, R2
FMT Z, "T", FXD * .Ø; WRT 13
      .
      .
      .
CONTINUE PROGRAM
```

69380A Sample Test Program

9830A Calculator

```
1Ø GOSUB 2ØØØ
2Ø DISP "ENTER PIN NO.";
3Ø INPUT A
4Ø A = 2 ↑ (A – 1)
5Ø GOSUB 24ØØ
6Ø CMD "?U7", "0ØØ4ØT"
7Ø FORMAT F1ØØ5.Ø
8Ø OUTPUT (13,7Ø) "F"A2"T";
9Ø DISP "TEST BIT"
1ØØ END
```

9820/21A Calculators

```
GSB "SPOLL"
ENT "ENTER PIN NO.", A
2 ↑ (A – 1)→A
GSB "OCT"
CMD "?U7", "0ØØ4ØT"
FMT Z, "F", FXD * .Ø; WRT 13, R2
FMT Z, "T", FXD * .Ø; WRT 13
DSP "TEST BIT"
END
```

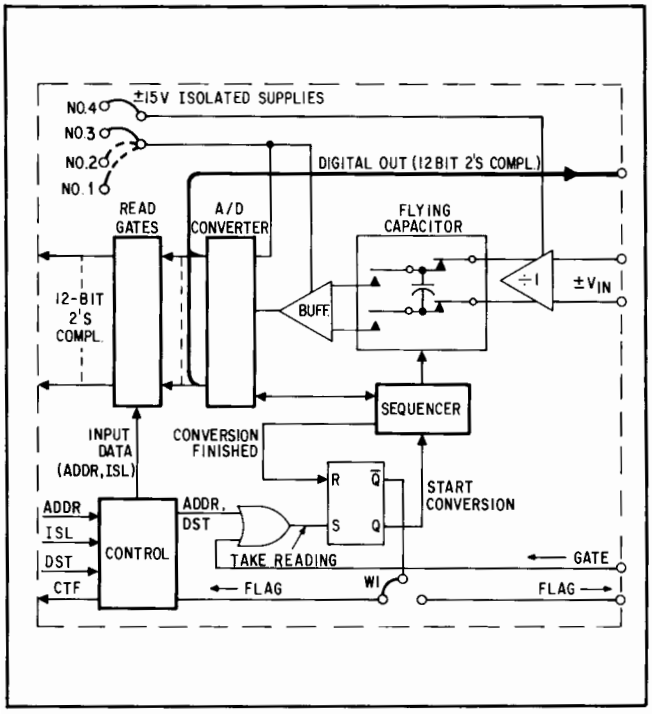
6-161 Voltage Monitor Card, 69421A

6-162 This input card monitors bipolar dc voltages in the range of +10.235 to −10.240 volts, and returns a 12-bit 2's complement digital word to indicate the magnitude and sign of the measured voltage. In addition an Option (102) is available which will allow voltage measurements between +102.35 and −102.40 volts. An explanation of the relationship between the applied voltages and corresponding return data words is given in Appendix A, Paragraph A-40. The following discussion will familiarize the user with programming the voltage monitor card 69421A. All examples will assume the voltage monitor card is installed in unit Ø, slot 4Ø5 (E), and a voltage regulator card. 69351A (which supplies the required ±15V bias voltages) is installed in unit Ø, slot 6ØØ.

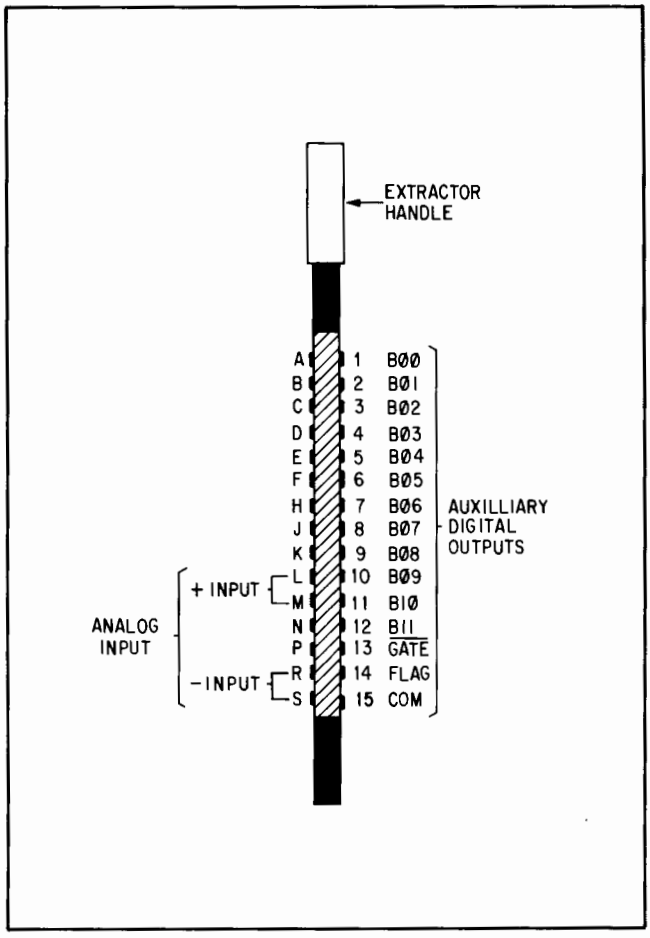
6-163 **Programming Example.** Example 34 illustrates measuring a voltage and storing the voltage, in octal form, in variable "A". Since it is convenient to work with decimal values when using this card, the octal to decimal subroutine (22ØØ) or "DEC") is inserted in line 5Ø.

6-164 In Example 34, a control word is first transmitted specifying input mode ("ISL" on), followed by an address word specifying a read with gate function (code "T"). When the voltage monitor card receives a gate, it will begin processing an analog voltage present at its input. At this point, the program will pause ("WAIT") for approximately 6 milliseconds to allow the voltage monitor card to convert the analog voltage to its digital equivalent.

6-165 An address word is then sent to the voltage monitor card specifying a read without gate function (code "X"). Subsequently addressing the multiprogrammer system to talk and the calculator to listen ("?5W") will allow an octal value representing the analog voltage to be transmitted to the bus and stored in variable (A).



69421A Block Diagram



69421A Connector

Example 34. Measuring a Voltage:

9830A Calculator

```
1Ø CMD "?U7", "ØØ24ØTET"
2Ø WAIT 6
3Ø CMD "?U7", "EX", "?5W"
4Ø ENTER (13, * ) A
5Ø GOSUB 22ØØ
```

9820/21A Calculators

```
CMD "?U7", "ØØ24ØTET"
"WAIT"
CMD "?U7", "EX", "?5W"
FMT * ; RED 13, A
GSB "DEC"
```

6-166 Timing Mode. Instead of programming the 6 millisecond "WAIT" in Example 34, TME could have been included in the control word. In this case, the card would not be read until the analog to digital conversion was complete. However, this would cause the multiprogrammer system to set its service request upon completion of the analog to digital conversion and would require doing a serial poll of the multiprogrammer system merely to reset its service request.

6-167 Sample Test Program. The sample test program for the 69421A card will allow the user to measure a voltage and display (or print) the value on the calculator. Of course, the program does not have to display (9830) or print (9820/21) the voltage and end as shown in the program but could use the voltage value calculated and stored in variable "A" and continue the program in any way desired. If an Option 102 card is being used, .005 in the eighth program line of example two should be changed to .05.

6-168 In the test program, the data returned from the voltage monitor card is converted to a decimal value by use of an octal to decimal subroutine (GOSUB 2200). If the decimal value exceeds 2047, it indicates a negative voltage and must be scaled before converting the value to a voltage. This is done by subtracting 4096 from the decimal value (A2 = A2 – 4096). The final decimal value is then converted to a voltage by multiplying the decimal value by the LSB, in volts, of the voltage monitor card (A = .005 * A2).

6-169 Notice that if the last program line was changed to "RETURN", instead of "END", the test program would comprise a voltage measuring subroutine which could easily be incorporated in a users program. Of course, the line numbers in the 9830 program would have to be changed, and the first line of the 9820/21 program would require a label (e. g., "MEAS").

6-170 Test Procedure. Perform the following steps when using the 69421A test program.

1. Load the program into the calculator, being sure to include the appropriate subroutines.

2. Apply the voltage to be measured between pins L and R (common) of the voltage monitor card.

3. Depressing RUN and EXECUTE on the 9830 calculator, or END, EXECUTE, and RUN PROGRAM on the 9820/21A calculators will allow the program to run and display (or print) the voltage on the calculator.

6-171 Isolated Digital Input Card, 69430A

6-172 This card allows the calculator to read 12-bits of logic level data that is isolated from ac earth ground, thereby eliminating ground loop problems in automatic test and control systems. The following discussion will familiarize the user with programming the isolated digital input card, 69430A. All examples will assume the input card is installed in unit 0, slot 412 (L).

6-173 Programming Example. Example 35 illustrates reading a return data word in octal form and storing it in variable "A". The data can be used in octal form or can be converted to decimal (see Paragraph A-50 in Appendix A) at the discretion of the user.

6-174 In Example 35, the isolated digital input card is read without a gate, as indicated by a gate code "X". This is necessary because there is no storage or CTF drive capability on this card. The only requirements for reading data from this card are: (1) A control word with ISL on must have been previously sent and (2) The card must be addressed.

6-175 Sample Test Program. The sample test program will enable the user to verify operation of 69430A Isolated

69421A Sample Test Program

9830A Calculator

```
10 CMD "?U7", "00240TET"
20 WAIT 6
30 CMD "?U7", "EX", "?5W"
40 ENTER (13, *) A
50 GOSUB 2200
60 IF A2 <= 2047 THEN 80
70 A2 = A2 –4096
80 A = .005 * A2
90 DISP A "VOLTS"
100 END
```

9820/21A Calculators

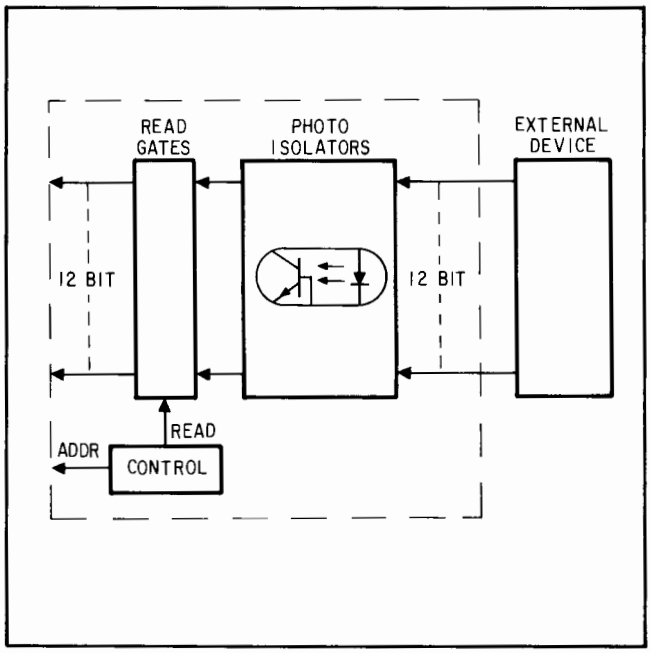
```
CMD "?U7", "00240TET"
"WAIT"
CMD "?U7", "EX", "?5W"
FMT * ; RED 13, A
GSB "DEC"
IF R2 ≤ 2047; JMP 2
R2 – 4096→R2
.005 * R2→A
FXD 3; PRT "VOLTAGE = ", A
END
```

Digital Input Cards equipped with Option 069, 084, 085, or 086. To verify the operation of 69430A cards equipped with Option 073, 087, 088, or 089, delete the third line of the sample test program. Options 073, 087, 088, and 089 have inverted data sense.

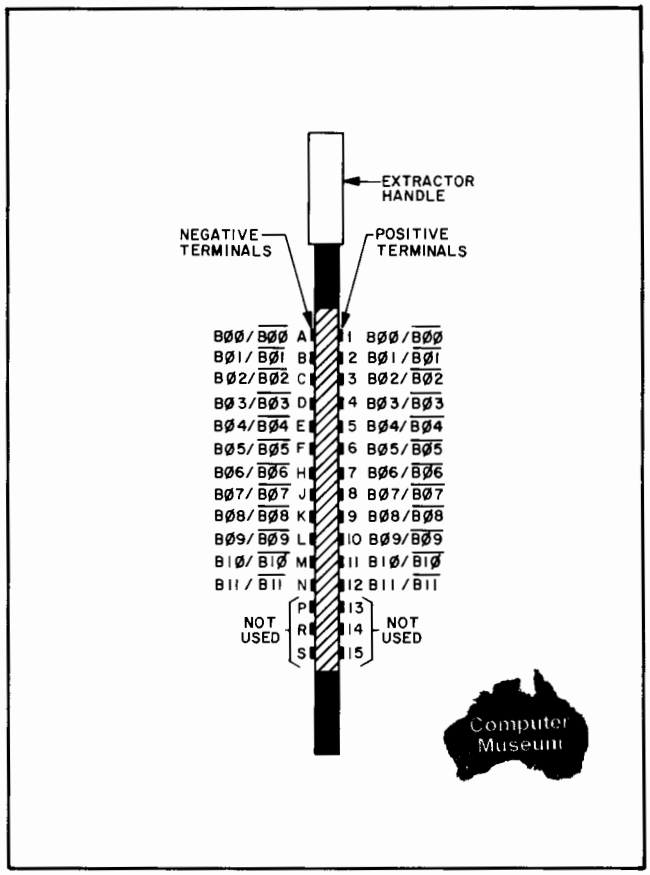
6-176 Test Procedure. Perform the following steps when using the 69430A test program.

1. Load the program into the calculator, being sure to include the appropriate subroutines.

2. From an external power supply, select a voltage level which corresponds to the high state of the option to be tested as specified in the Isolated Digital Input Card Manual. (e. g., 5 volts would suffice for Options 069 and 073).



69430A Block Diagram



69430A Input Connector

Example 35. Reading an Isolated Digital Card:

9830A Calculator

```
110 CMD "?U7", "00240TLX", "?5W"
120 ENTER (13, * )A
      .
      .
CONTINUE PROGRAM
```

9820/21A Calculators

```
CMD "?U7", "00240TLX", "?5W"
FMT * ; RED 13, A
      .
      .
CONTINUE PROGRAM
```

69430A Sample Test Program

9830A Calculator

```
10 CMD "?U7", "00240TLX", "?5W"
20 ENTER (13, * ) A
30 A = 7777 – A
40 GOSUB 2200
50 GOSUB 2600
60 END
```

9820/21A Calculators

```
CMD "?U7", "00240TLX", "?5W"
FMT * ; RED 13, A
7777 – A→A
GSB "DEC"
GSB "BIT"
END
```

3. Apply an input voltage to the input connector of the isolated input board as follows:

a. Apply the positive side of the input voltage to any pin, 1 through 12, selected by the user.

b. Apply the negative side of the input voltage to a lettered pin which corresponds to the numbered pin previously selected. (e. g., pin A corresponds to pin 1 and pin N corresponds to pin 12).

4. Depressing RUN and EXECUTE on the 9830A calculator or END, EXECUTE, and RUN PROGRAM on 9820/21A calculators will allow the program to run. The calculator will stop with the input bit and pin number displayed (9830) or printed (9820/21). If no data has been input the calculator will stop with NO DATA RECEIVED displayed.

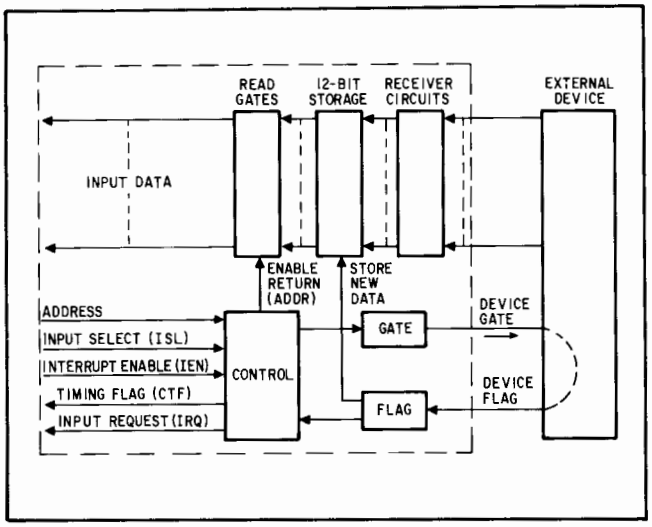
5. Repeating steps three and four will enable the user to check each data input line on the card.

6-177 Digital Input Card, 69431A

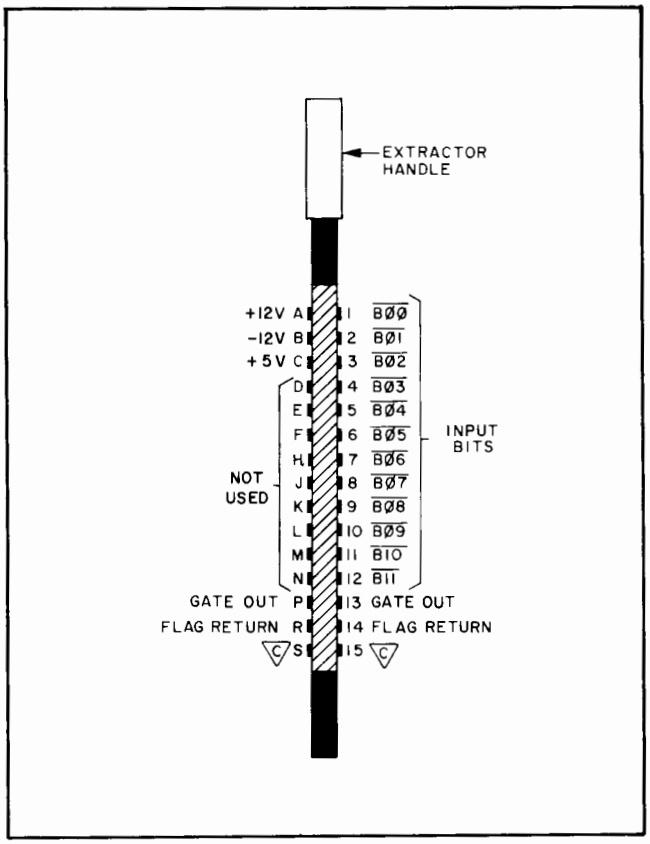
6-178 This card allows the calculator to read 12-bits of logic level or contact closure data that is referenced to the logic common (ac earth ground). Gate/flag circuits are provided on each 69431A to allow external devices to control the service request of the bus through the multiprogrammer interface. The following discussion will familiarize the user with programming the digital input card 69431A. All examples will assume the digital input card is installed in unit Ø, slot 4Ø7 (G).

NOTE

When using this card without an external gate/flag timing circuit, the card's gate output must be jumpered to its flag input (see Paragraph 6-51).



69431A Block Diagram



69431A Input Connector

6-179 Operating Modes. Basically there are three modes of operation which apply to this card:

1. Automatic Handshake Mode (TME off): Used to read data from an input card when it is not necessary to handshake with an external device.

2. Timing Mode (TME on): Also called the dedicated input mode, used to read data from an input card when it is necessary for the card to handshake with an external device.

3. Interrupt Mode: Useful for activating many input cards simultaneously, then reading back data after a handshake is completed between an external device and one of the activated input cards.

6-180 Regardless of the mode used, the return data word is input in octal form to variable "A". The data can be used in octal form or can be converted to decimal (see Paragraph A-50 in Appendix A) at the discretion of the user.

6-181 Automatic Handshake Mode. Example 36 illustrates reading back data in the automatic handshake mode (TME off). This mode is used when a handshake between the 69431A card and external device (data source) is not required. Note in Example 36 that IRQ is subtracted from the return data word in the third program so that the data can be utilized. Checking and subtracting IRQ is explained in Paragraph 6-46.

Example 36. Reading Data in Automatic Handshake Mode (TME off):

9830A Calculator

```
110 CMD "?U7", "00240TGT", "?5W"
120 ENTER (13, *) A
130 A = A – 10000
              .
              .
CONTINUE PROGRAM
```

9820/21A Calculators

```
CMD "?U7", "00240TGT", "?5W"
FMT *; RED 13, A
A – 10000→A
              .
              .
CONTINUE PROGRAM
```

Example 37. Reading Data in Timing Mode (TME on):

9830A Calculator

```
110 CMD "?U7", "00260T"
120 GOSUB 2000
130 CMD "?U7", "GT"
              .
              .
200 IF STAT 13 > 1 THEN 500
210 GOSUB 2000
220 IF A4 # 64 THEN 500
              .
              .
230 CMD "?5W"
240 ENTER (13, *) A
250 A = A – 10000
              .
              .
500 REM CONTINUE PROGRAM
```

9820/21A Calculators

```
CMD "?U7", "00260T"
GSB "SPOLL"
CMD "?U7", "GT"
              .
              .
IF RDS 13 > 1; GTO "CONT"
GSB "SPOLL"
IF R4 ≠ 64; GTO "CONT"
              .
              .
CMD "?5W"
FMT *; RED 13, A
A – 10000→A
              .
              .
"CONT" CONTINUE PROGRAM
```

6-182 Timing Mode. Example 37 illustrates reading back data in the timing mode (TME on). This technique can be used to read digital information from an external device which first requires the input card to handshake with the device. When addressed with a gate the card will initiate a gate to the external device and will not read data until it receives a flag from the device indicating data is ready. Pin 13 should be connected to the gate input and pin 14 connected to the flag output of the external device. When the input card is addressed with a gate, it will initiate a gate to the external device. When the flag from the external device is returned to the input card, the card will read in the data and the multiprogrammer system will set SRQ to indicate the card has data ready.

6-183 Interrupt Mode. Example 38 illustrates the interrupt mode of operation. Programming the digital input card in the interrupt mode consists of three basic steps:

1. The card must be armed using either one of the following methods:

a. Sending and gating ("T") a control word with ISL on, followed by an address word containing the card address and a gate code "T", then another control word accompanied by the "T" gate code with IEN and TME on.

b. If W6 jumper is installed, simply send and gate ("T" code) a control word with IEN and TME on.

2. When an interrupt occurs, the bus service request line will be set. In this case, an interrupt occurs when the external device returns the trailing edge of flag. A poll of the previously armed cards must then be done by addressing the cards in the input mode (ISL on), reading the card without a gate ("X" code) and examining the state of IRQ. If the input data word is equal to or greater than 10000 it indicates the card had interrupted and has data ready.

3. The card should now be disarmed by sending a control word with ISL off, followed by an output word containing the card address and a gate code "T".

6-184 Example 38 assumes that jumper W6 has been removed from the input card, allowing the user to selectively arm input cards. Pins 13 and 14 of the input card should be connected to the gate and flag lines of the external

device as previously noted.

NOTE

If jumper W6 is installed on input card(s) it will allow the input card(s) to be activated whenever a control word containing IEN is transmitted along with gate code "T". This precludes selectively arming input cards which could cause programming problems if group arming is not understood (see Paragraph 6-54).

6-185 The first input card that receives a flag from its external device will set SRQ. Therefore, it is very critical to test the state of IRQ before extracting data from the input card, as any activated card could have interrupted and set SRQ. If IRQ is on (input word equal to or greater than 10000) the data can be extracted by subtracting 10000 from the input word. If IRQ is off (input word less than 10000) the card does not have data ready. It is possible to read data from any input card in an effort to find out which card caused SRQ to be set (see Paragraph 6-53). The card should be read without a gate ("X" gate code) during the IRQ polling routine. The input card would begin a new transfer sequence if a "T" gate code were sent.

6-186 Sample Test Program. The sample test program will enable the user to verify operation of 69431A cards equipped with Options 069 or 090. To verify the operation of 69431A cards equipped with Option 073 (opposite logic sense), change the seventh line of the sample program as follows:

9830 Calculator: A = 17777 – A
9820/21 Calculators: 17777 –A→A

6-187 Test Procedure. Perform the following steps when using the 69431A sample test program.

1. Load the program into the calculator, being sure to include the appropriate subroutines.

2. On the input connector of the digital input card, temporarily install a jumper between pins 13 and 14, and another jumper between pin 15 and any user selected input pin (1 through 12).

3. Depressing RUN and EXECUTE on the 9830 calculator or END, EXECUTE and RUN PROGRAM on 9820/21 calculators will allow the program to run. If IRQ is not operating proprerly, or if a flag is not returned, the program will be terminated and the calculator will display (9830) or print (9820/21) "IRQ OR FLAG RETURN ERROR". This problem should be corrected before rerunning the test. If IRQ is operating properly and the flag is returned normally, the calculator will stop with the input bit and pin number displayed (or printed). If no data line (pins 1-12) has been jumpered to pin 15, the calculator will stop with "NO DATA RECEIVED" displayed.

4. Despressing the CONT and EXECUTE (or the RUN PROGRAM) keys after the bit has been tested will cause the

Example 38. Interrupt Mode:

9830A Calculator

```
110 CMD "?U7","00240TGT00460T"
                .
                .
                .
200 IF STAT 13>1 THEN 500
210 GOSUB 2000
220 IF A4#64 THEN 500
                .
                .
                .
230 CMD "?U7","00240TGX","?5W"
240 ENTER (13,*) A
250 IF A<10000 THEN 500
260 A=A-10000
270 CMD "?U7","00040TGT"
                .
                .
                .
500 REM CONTINUE PROGRAM
```

9820/21A Calculators

```
CMD "?U7","00240TGT00460T"
                .
                .
                .
IF RDS 13>1;GTO "CONT"
GSB "SPOLL"
IF R4≠64;GTO "CONT"
                .
                .
                .
CMD "?U7","00240TGX","?5W"
FMT *;RED 13,A
IF A≤9999;GTO "CONT"
A-10000→A
CMD "?U7","00040TGT"
                .
                .
                .
"CONT" CONTINUE PROGRAM
```

9830A Calculator

```
10 GOSUB 2000
20 CMD "?U7", "00240TGT", "?5W"
30 ENTER (13, * ) A
40 IF A = 10000 OR A > 10000 THEN 70
50 DISP "IRQ OR FLAG RETURN ERROR"
60 GO TO 180
70 A = A – 10000
80 GOSUB 2200
90 GOSUB 2600
100 STOP
110 DISP "TME TEST: REMOVE JUMPERS"
120 STOP
130 CMD "?U7", "00260T"
140 GOSUB 2000
150 CMD "?U7", "GT00240T"
160 GOSUB 2000
170 DISP "TEST COMPLETE"
180 END
```

9820/21A Calculators

```
GSB "SPOLL"
CMD "?U7", "00240TGT", "?5W"
FMT * ; RED 13, A
IF A > 9999; JMP 3
PRT "IRQ OR FLAG", "RETURN ERROR"
JMP 12
A – 10000→A
GSB "DEC"
GSB "BIT"
STP
PRT "TME TEST", "REMOVE JUMPERS"
STP
CMD "?U7", "00260T"
GSB "SPOLL"
CMD "?U7", "GT00240T"
GSB "SPOLL"
DSP "TEST COMPLETE"
END
```

calculator to display (or print) "TME TEST: REMOVE JUMPERS".

5. Disconnect all jumpers.

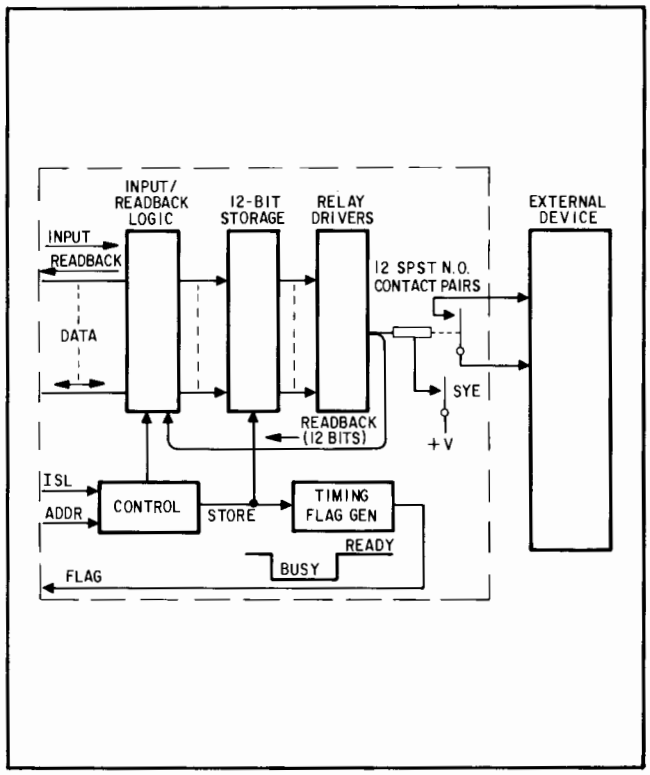
6. Press CONT and EXECUTE (or RUN PROGRAM). The calculator should hang-up .

7. Momentarily short pins 13 and 14 together. The calculator will display "TEST COMPLETE".

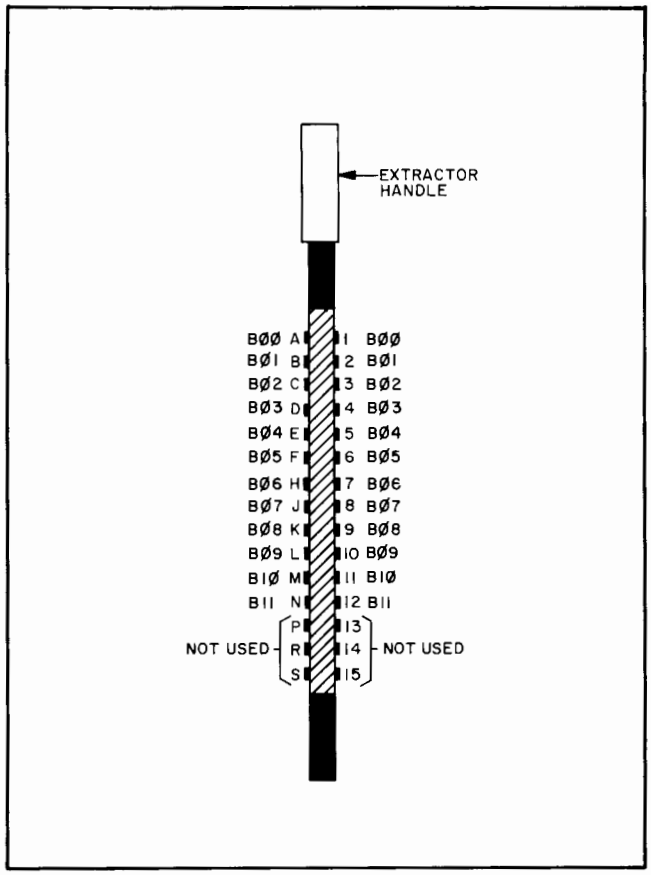
6-188 Relay Output With Readback, 69433A

6-189 This card provides 12, independent, SPST, normally open (form A) relays. Output data will be in the form of contact closures. In addition, the 69433A relay output card allows the calculator to examine the status of the relay coil drive circuits on the card, before and/or after the contacts are actually changed. The following discussion will familiarize the user with programming the relay output/readback card 69433A. All examples will assume the card is installed in unit 0, slot 404 (D).

6-190 Programming Output Data. Examples 39 and 40 illustrate programming output data from a relay output/readback card. A control word is first sent containing SYE only, followed by a data word containing the card slot address, and the output data desired. Although this card returns a CTF flag, as a function of internal timing circuits, TME is not included in the control words of the programming examples. This is because the statement execution time of 9820/21 and 9830 calculators is greater than the period of the internal timing circuits. If TME were included, the program would run essentially the same. However, the service request would be set whenever the card finished processing its output data. This would necessitate doing a serial poll of the multiprogrammer system merely to reset its service request.



69433A Block Diagram



69433A Output Connector

6-191 Example 39 illustrates using a data constant of 7777_8 to program a relay output card located in slot 4Ø4 (D).

6-192 In some cases it might be desirable to specify the output data in the form of a variable. The data could then be changed as a function of the user's program. Example 40 illustrates using a data variable ("A2" or "R2") to program a relay output card. It should be kept in mind that this must be an octal value. If desired, the decimal to octal subroutine provided in Appendix A may be used to enable programs to be written with decimal values.

6-193 Reading Relay Status. The user may examine the status of the relays at any time. The relay output/readback card requires a maximum of 4 milliseconds to program an output. No attempt should be made to read the card within 4 milliseconds after data has been programmed. Since a program line for the 9820, 9821, and 9830 calculators requires more than 4 milliseconds to execute, it is permissable to read the card in a statement immediately following the statement which programs an output. Faster calculators would require the use of TME, status checks, and a serial poll as previously discussed in general programming information (see Paragraph 6-48).

6-194 In Example 41 the return data word is readback in octal form to variable "A". It is then up to the user to process the data. For example, the data could be used in octal form, or converted to decimal by the octal to decimal subroutine in Appendix A.

6-195 Sample Test Program. The sample test program will enable the user to verify operation of the 69433A cards.

6-196 Test Procedure. Perform the following steps when using the 69433A sample test program.

1. Load the program into the calculator, being sure to include the appropriate subroutines.

2. Depressing RUN and EXECUTE on the 9830 calculator or END, EXECUTE and RUN PROGRAM on 9820/21 calculators will allow the program to run.

3. The calculator will stop with "ENTER PIN NO." displayed. The user must enter a pin number from 1 to 12, then depress EXECUTE on the 9830 calculator or RUN PROGRAM on 9820/21 calculators.

4. The calculator will stop with "TEST BIT" displayed. At this point, the user should use an ohmmeter to check the contact resistance of the relays. Relay outputs can be measured between a numbered pin, 1 through 12, and a corresponding lettered pin (e. g., 1 and A, 2 and B, 15 and S). All relay contacts should be open except for the contacts corresponding to the pin number that was entered in step 3.

5. On 6940 multiprogrammer, place data source switch to LOCAL. Momentarily depress CLEAR REGISTER

Example 39. Programming Relay Output Using Data Constants:

9830A Calculator

```
11Ø  CMD  "?U7" , "ØØ4ØTD7777T"
          .
          .
          .
CONTINUE PROGRAM
```

9820/21A Calculators

```
CMD  "?U7" ,"ØØ4ØTD7777T"
          .
          .
          .
CONTINUE PROGRAM
```

Example 40. Programming Relay Output Using Data Variables:

9830A Calculator

```
110 CMD "?U7", "00040T"
120 FORMAT F1005.0
130 OUTPUT (13,120) "D"A2"T";
                .
                .
                .
CONTINUE PROGRAM
```

9820/21A Calculators

```
CMD "?U7", "00040T"
FMT Z, "D", FXD * .0; WRT 13, R2
FMT Z, "T", FXD * .0; WRT 13
                .
                .
                .
CONTINUE PROGRAM
```

Example 41. Reading Relay Status:

9830A Calculator

```
140 CMD "?U7", "00240TDX", "?5W"
150 ENTER (13, * ) A
```

9820/21A Calculators

```
CMD "?U7", "00240TDX", "?5W"
FMT * ; RED 13, A
```

69433A Sample Test Program

9830A Calculator

```
10 GOSUB 2000
20 DISP "ENTER PIN NO.";
30 INPUT A
40 A = 2 ↑ (A – 1)
50 GOSUB 2400
60 CMD "?U7", "00040T"
70 FORMAT F1005.0
80 OUTPUT (13,70) "D"A2"T";
90 DISP "TEST BIT"
100 STOP
110 CMD "?U7", "00240TDX", "?5W"
120 ENTER (13, * ) A
130 GOSUB 2200
140 GOSUB 2600
150 END
```

9820/21A Calculators

```
GSB "SPOLL"
ENT "ENTER PIN NO.", A
2 ↑ (A – 1)→A
GSB "OCT"
CMD "?U7", "00040T"
FMT Z, "D", FXD * .0; WRT 13, R2
FMT Z, "T", FXD * .0; WRT 13
DSP "TEST BIT"
STP
CMD "?U7", "00240TDX", "?5W"
FMT * ; RED 13, A
GSB "DEC"
GSB "BIT"
END
```

switch. Press switches 15-12 (control word) and switch 5 (SYE). Depress LOAD OUTPUT switch.

6. Depress CLEAR REGISTER switch. Press switch 14, then select and press a data switch from 0 to 11. Depress LOAD OUTPUT switch.

7. Switch data source switch back to REMOTE.

8. Depress CONT and EXECUTE on the 9830 calculator or RUN PROGRAM on 9820/21 calculators. The calculator will display the bit (and corresponding pin number) that was manually programmed in step 6.

6-197 Event Sense Card, 69434A

6-198 This card monitors up to 12 external contact closures and sets the bus service request when one or more contacts change for 20 milliseconds or longer with respect to 12 reference bits stored on the card. The following discussion will familiarize the user with programming the event sense card 69434A. All examples will assume the event sense card is installed in unit 0, slot 402 (B).

6-199 Return Data Word. The return data word will be input in octal form to variable "A". It will then be up to the user to process the data. For example, the data could

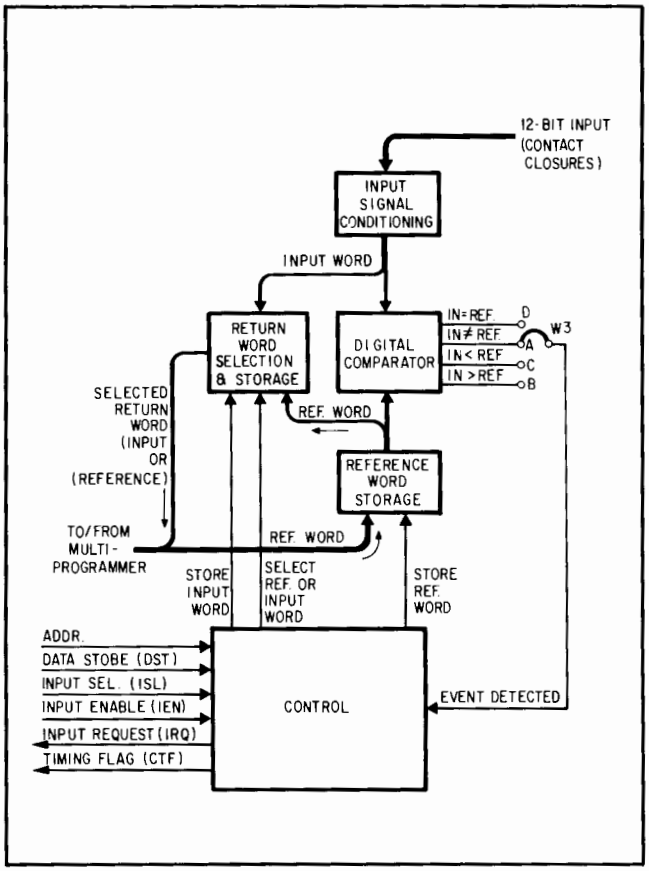
be used in octal form or converted to decimal by the octal to decimal subroutine in Appendix A, (Paragraph A-50).

6-200 Interrupt Mode. Basically, this card is only used in the interrupt mode, either singly or with other cards having interrupt capability, and will set the bus service request when any one of four conditions prevails. The interrupting condition is selected by jumper (W3) as follows:

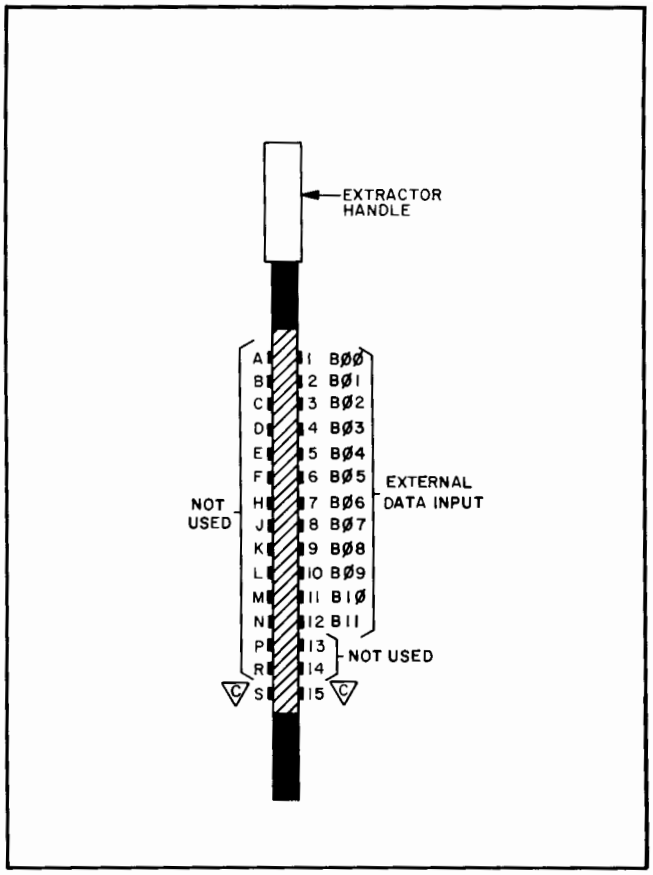
*W3 Position	Interrupting Condition
A	Input Word = Reference Word
B	Input Word > Reference Word
C	Input Word < Reference Word
D	Input Word ≠ Reference Word

* The card is shipped from the factory with the W3 jumper in the D position.

6-201 The event sense card continually compares the input word with a reference word while the system is in the interrupt mode and stores the entire input word at the moment the event (interrupt) occurs. Both the interrupting input word and the reference word can be read back after the event has occurred. As shipped from the factory, the event sense card will not store input data unless the multi-



69434A Block Diagram



69434A Input Connector

programmer is in the interrupt mode with the gate set. The 69436A process interrupt card (see Paragraph 6-220) differs in that it will store input transitions regardless of the system mode, but does not make comparisons with a reference word and will alter the original interrupting word in storage if additional transitions occur.

6-202 As part of the total transfer sequence, the event sense card is also operated in the automatic handshake mode (both input and output). Programming the card consists of four basic steps.

1. Loading the reference word initializes the card.

2. Arming permits the card to control the flag and thus set the bus service request.

3. Reading the card returns the interrupting word ("X", read without gate code) and the reference word ("T", read with gate code) to the calculator.

4. Disarming the card terminates the transfer operation.

6-203 Loading Reference Word. Example 42 illustrates loading a fixed reference word represented by 7777_8 into storage on the event sense card. Notice that a control word containing SYE only is sent first. Although SYE has no

effect on this card, it is included in the control word to avoid disabling the outputs of previously programmed cards.

6-204 In some cases it might be desirable to specify the reference word as a variable. Example 43 illustrates loading a reference word as a variable. In Example 43, "A2" (or "R2" when using 9820/21A calculators) is the variable which specifies the reference word. It should be kept in mind that this must be an octal value. If desired, the decimal to octal subroutine provided in Appendix A may be used to enable programs to be written with decimal values.

6-205 Arming and Reading the Card. Once the reference word is loaded, the card must be armed before it can generate an interrupt and set the bus service request. The event sense card may be armed as follows:

1. Sending a control word with ISL on, followed by an address word containing the card address and a gate code "T", then another control word with IEN and TME,

or

2. If the W6 jumper is installed simply sending and gating a control word with IEN and TME on.

NOTE

If the W6 jumper is installed on an event sense card, it will allow the card to be armed whenever a control word containing IEN is gated. This precludes selectively arming the card or any other card with W6 installed. This method may cause programming problems if used indiscriminately (see Paragraph 6-54).

6-206 Example 44 illustrates selectively arming a card. It is assumed that the W6 jumper has been removed.

6-207 The second program lines, shown in Example 44, test to determine whether the service request has been set, while the fourth program lines test to determine if the multiprogrammer system is the bus device that requested service. The seventh program lines examine the IRQ bit of the addressed card (in this case the 69434A card in slot 402 (B). If the event sense card generated the service request, then the IRQ bit is subtracted from the return data word and the new value is stored in variable "A". After the input word is read and stored in variable "A", the card is again addressed (ISL is still on) and a "T" (read with gate code) is sent to the card. This causes the reference word to be sent to the multiprogrammer interface from which it is read and stored in variable "B".

NOTE

The event (interrupting input word) is lost from storage when a "T" code is sent to the 69434A card after an interrupt has occurred. Therefore, the input word must first be read without a gate and then a "T" code sent and the reference word read. A jumper (W2) permits the user to modify the manner in which the input word is stored. Consult the 69434A card operating and service manual for details.

6-208 Disarming the Card. Once the card has generated an interrupt and been read, it should be disarmed (the event sense card cannot be recycled). Example 45 illustrates disarming the card. The card is disarmed whenever it is addressed and gated with a "T" code in the output mode. Therefore, the user may simultaneously disarm the card and load in a new reference word after which the card may be rearmed at the user's convenience (either immediately or

Example 42. Loading a Reference Word Using Data Constants:

9830A Calculator

```
110 CMD "?U7", "00040B7777T"
```

9820/21A Calculators

```
CMD "?U7", "00040B7777T"
```

Example 43. Loading a Reference Word Using Data Variables:

9830A Calculator

```
110 CMD "?U7", "00040T"
120 FORMAT F1005.0
130 OUTPUT (13,120) "B"A2"T";
                .
                .
                .
CONTINUE PROGRAM
```

9820/21A Calculators

```
CMD "?U7", "00040T"
FMT Z, "B", FXD * .0; WRT 13, R2
FMT Z, "T", FXD * .0; WRT 13
                .
                .
                .
CONTINUE PROGRAM
```

at some later time).

6-209 Sample Test Program. The sample test program will enable the user to verify operation of 69434A cards.

6-210 Test Procedure. Perform the following steps when using the 69434A sample test program.

1. Load the program into the calculator, being sure to include the appropriate subroutines.

2. Connect a jumper between pin 15 and any pin 1 through 12 on the card edge connector.

3. Depressing RUN and EXECUTE on the 9830A calculator or END, EXECUTE, and RUN PROGRAM on the 9820/21A calculators will allow the program to run.

4. The calculator will print (9820/21A) or display (9830A) the bit and pin number selected.

5. Depress CONT and EXECUTE (9830A) or RUN PROGRAM (9820/21A) and the calculator will display "ENTER BIT NUMBER". The bit previously selected should be entered and EXECUTE (9830A) or RUN PROGRAM (9820/21A) depressed. If an incorrect bit number is entered, the calculator will display "WRONG BIT". The program may be continued by returning to the beginning of step 5.

6. The calculator will halt with the GATE indicator on the 59500A lit. Remove the jumper installed in step 2 and the calculator will display "TEST COMPLETE" Repeating steps 2 through 6 will allow the user to check operation of all twelve input lines.

Example 44. Arming and Reading an Event Sense Card:

9830A Calculator

```
140 CMD "?U7", "00240TBT00460T"
                .
                .
                .
200 IF STAT 13 > 1 THEN 500
210 GOSUB 2000
220 IF A4 # 64 THEN 500
                .
                .
                .
230 CMD "?U7", "00240TBX", "?5W"
240 ENTER (13, * ) A
250 IF A < 10000 THEN 500
260 A = A - 10000
270 CMD "?U7", "BT", "?5W"
280 ENTER (13, * ) B
290 B = B - 10000
                .
                .
                .
500 REM CONTINUE PROGRAM
```

9820/21A Calculators

```
CMD "?U7", "00240TBT00460T"
                .
                .
                .
IF RDS 13 > 1; GTO "CONT"
GSB "SPOLL"
IF R4 ≠ 64; GTO "CONT"
                .
                .
                .
CMD "?U7", "00240TBX", "?5W"
FMT * ; RED 13, A
IF A ≤ 9999; GTO "CONT"
A - 10000→A
CMD "?U7", "BT", "?5W
FMT * ; RED 13, B
B - 10000→B
                .
                .
                .
"CONT" CONTINUE PROGRAM
```

Example 45. Disarming the Event Sense Card:

9830A Calculator

```
300 CMD "?U7", "00040TBT"
                .
                .
                .
500 REM CONTINUE PROGRAM
```

9820/21A Calculators

```
CMD "?U7", "00040TBT"
                .
                .
                .
"CONT" CONTINUE PROGRAM
```

9830A Calculator

```
10 GOSUB 2000
20 CMD "?U7", "00040TB7777T00240TBT00460T"
30 IF STAT 13 > 1 THEN 30
40 GOSUB 2000
50 CMD "?U7", "00240TBX", "?5W"
60 ENTER (13, * )A
70 B = A = 7777 - (A-10000)
80 GOSUB 2200
90 GOSUB 2600
100 CMD "?U7", "00040TBT"
110 STOP
120 DISP "ENTER BIT NUMBER";
130 INPUT C
140 A = 2↑C
150 GOSUB 2400
160 IF A2 = B THEN 200
170 DISP "WRONG BIT"
180 STOP
190 GOTO 120
200 CMD "?U7", "00040T"
210 FORMAT F1005.0
220 B = 7777 - B
230 OUTPUT (13,210)"B"B"T";
240 CMD "?U7", "00240TBT00460T"
250 IF STAT 13 > 1 THEN 250
260 GOSUB 2000
270 CMD "?U7", "00040TBT"
280 DISP "TEST COMPLETE"
290 END
```

9820/21A Calculators

```
GSB "SPOLL"
CMD "?U7", "00040TB7777T00240TBT00460T"
IF RDS 13 > 1; JMP 0
GSB "SPOLL"
CMD "?U7", "00240TBX", "?5W"
FMT * ; RED 13, A
7777 - (A - 10000)→A→B
GSB "DEC"
GSB "BIT"
CMD "?U7", "00040TBT"
STP
ENT "ENTER BIT NO.", C
2 ↑ C→A
GSB "OCT"
IF R2 = B; JMP 4
DSP "WRONG BIT"
STP
JMP-6
CMD "?U7", "00040T"
7777 - B→B
FMT Z, "B", FXD * .0; WRT 13, B
FMT Z, "T", FXD * .0; WRT 13
CMD "?U7", "00240TBT00460T"
IF RDS 13 > 1; JMP 0
GSB "SPOLL"
CMD "?U7", "00040TBT"
DSP "TEST COMPLETE"
END
```

6-211 Pulse Counter Card, 69435A

6-212 This card will count pulses, up or down, in the range of 0 to 4095. A carry or borrow pulse is generated as the count goes above 4095 or below 0. These pulses allow multiple counter cards to be cascaded for greater counting capability or they can serve as alarm signals. The card can also be used as a pre-set counter. The following discussion will familiarize the user with programming the pulse counter card 69435A. All examples will assume the pulse counter card is installed in unit 0, slot 413 (M).

6-213 Programming Examples. Programming the pulse counter card consists of two basic steps:

1. The pulse counter card must first be preset by sending a control word to the multiprogrammer with ISL off, then sending a data word containing the slot address of the card and the data value which the counter is to be preset to. At this point, the card is always addressed with a gate, as indicated by a gate code "T". This is necessary to preset the counter.

2. The pulse counter card can be read by sending a control word with ISL on, then addressing the card with a read without gate command. The return data word will be readback in octal form to variable "A". It will then be up to the user to process the data. For example, the data could be used in octal form or converted to decimal by the octal to decimal subroutine in Appendix A.

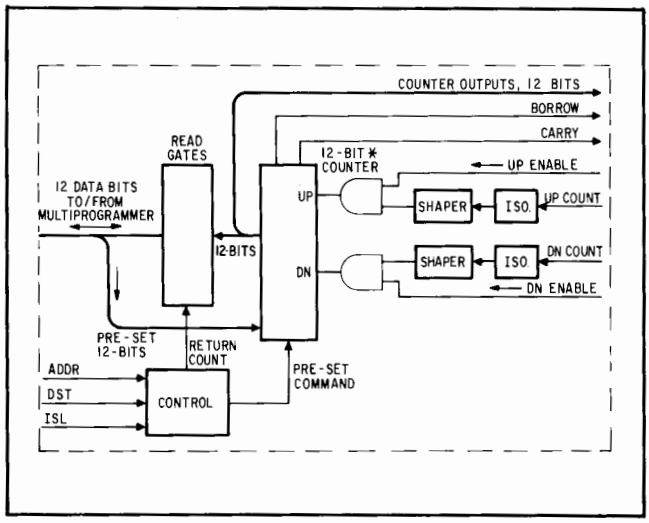
6-214 In Example 46 it is assumed the card is to be used as a positive counter (the pulses are entered in the count-up inputs) and it is desired to preset the counter to zero. First a control word with ISL off is sent, followed (in the same statement) by the card address and a data value of 0 (MT or M0000T are both acceptable). Later in the program, the count contained in the card is read and stored in a variable "A". Of course, if no pulses were entered into the card in the interval, "A" would equal the preset value.

6-215 The pulse counter card can be preset to any value desired, up to its maximum of 4095. But the value must be converted to octal (see Appendix A, Paragraph A-21) before being sent out to the multiprogrammer system. For example, if it were desired to preset the counter to 4095 and use the card to count down, the first program line of Example 46 could be changed to read as follows:

```
CMD "?U7", "00040TM7777T"
```

6-216 To use the card to count down, input pulses are applied to the count-down inputs of the card. The value read into "A" would be the octal equivalent of decimal 4095 minus the number of pulses received.

6-217 In some cases it might be desirable to use a variable to preset the counter to a value which could change as a function of the user's program. Example 47 illustrates the format required to output variables. In this example, "A2" (or "R2" when using a 9820/21) is the variable that is sent out to preset the pulse counter card. It should be kept in mind that this must be an octal value. If desired, the decimal to octal subroutine provided in Appendix A may be used to enable programs to be written with decimal values.



69435A Block Diagram

Example 46. Programming a Pulse Counter Card with a Preset Value Using Data Constants:

9830A Calculator

```
110 CMD "?U7", "00040TMT"
                .
                .
                .
200 CMD "?U7", "00240TMX", "?5W"
210 ENTER (13, *) A
                .
                .
                .
CONTINUE PROGRAM
```

9820/21A Calculators

```
CMD "?U7", "00040TMT"
                .
                .
                .
CMD "?U7", "00240TMX", "?5W"
FMT *; RED 13, A
                .
                .
                .
CONTINUE PROGRAM
```

Example 47. Programming a Pulse Counter Card with a Preset Value Using Data Variables:

9830A Calculator

```
110 CMD "?U7", "00040T"
120 FORMAT F1005.0
130 OUTPUT (13, 120) "M"A2"T";
                .
                .
                .
200 CMD "?U7", "00240TMX", "?5W"
210 ENTER (13, *)A
                .
                .
                .
CONTINUE PROGRAM
```

9820/21A Calculators

```
CMD "?U7", "00040T"
FMT Z, "M", FXD *.0; WRT 13, R2
FMT Z, "T", FXD *.0; WRT 13
                .
                .
                .
CMD "?U7", "00240TMX", "?5W"
FMT *; RED 13, A
                .
                .
                .
CONTINUE PROGRAM
```

6-218 Sample Test Program. This program will enable the user to verify the operation of pulse counter cards 69435A. Jumpers W1 and W2 must not be installed when running this program.

6-219 Test Procedure. Perform the following steps when using the 69435A sample test program.

1. Load the desired program into the calculator, being sure to include the appropriate subroutines.

2. On the input connector of the pulse counter card, install a jumper between pins F and 15.

3. Connect a second jumper as follows:

a. For a count-up test: connect a jumper between pin E of the pulse counter card and test point GAT 300 located on the logic and timing card in slot 300.

or

b. For a count-down test: connect a jumper between pin H of the pulse counter card and test point GAT 300 located on the logic and timing card in slot 300.

4. Depressing RUN and EXECUTE on a 9830 calculator or END, EXECUTE, and RUN PROGRAM on 9820/21 calculators will allow the program to run and the calculator will stop with "BEGIN COUNTING" displayed.

5. On 6940 multiprogrammer, place DATA SOURCE switch to LOCAL. Momentarily depress CLEAR REGISTER switch. Press (light) switches 15-12 (control word) and switch 5 (SYE).

6. Press LOAD OUTPUT switch as many times as desired. (Pressing the LOAD OUTPUT switch will generate a data strobe pulse which we will count.)

7. Switch DATA SOURCE switch back to REMOTE.

8. Depress the CONT and EXECUTE (or RUN PROGRAM for 9820/21) keys on the calculator.

a. If a count up test is being run, the calculator will display (or print when using 9820/21 calculators): count = (the number of times LOAD OUTPUT switch has depressed in step 6).

b. If a count down test is being run the calculator will display (or print): count = (4094 minus the number of times the LOAD OUTPUT switch was depressed in step 6).

6-220 Process Interrupt Card, 69436A

6-221 This card provides TTL and open collector compatible edge detectors; one positive and one negative for each of 12 storage latches. Logic transitions lasting 100 nanoseconds or longer are detected, stored, and used to control the HP-IB service request through the 59500A multiprogrammer interface. The following discussion will familiarize the user with programming the process interrupt card 69436A. All examples will assume the process interrupt card is installed in unit 0, slot 403 (C).

6-222 Return Data Word. The return data word will be input in octal form to variable "A". It will then be up to the user to process the data. For example, the data could be used in octal form or converted to decimal by the octal to decimal subroutine in Appendix A, (Paragraph A-50).

6-223 Interrupt Mode. Basically this card is only used in the interrupt mode, either singly or with other cards having interrupt capability, and will set the bus service request when any one of the 24 edge detectors is toggled. As part of the total transfer sequence, however, the card is also operated in the automatic handshake mode (both input and output). Programming the process interrupt card consists of five basic steps.

1. Initializing: Sets all edge detectors to a known (reset) state.

2. Arming: Permits the card to control the flag and thus set the service request.

3. Reading: returns data to the calculator.

69435A Sample Test Program

9830A Calculator

```
10 CMD "?U7", "00040TMT"
20 DISP "BEGIN COUNTING"
30 STOP
40 CMD "?U7", "00240TMX", "?5W"
50 ENTER (13, *) A
60 GOSUB 2200
70 DISP "COUNT = "A2-1
80 END
```

9820/21A Calculators

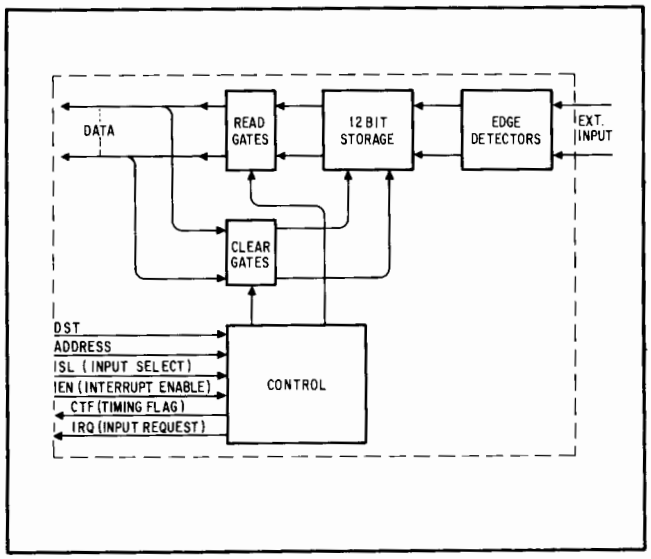
```
CMD "?U7", "00040TMT"
DSP "BEGIN COUNTING"
STP
CMD "?U7", "00240TMX", "?5W"
FMT *; RED 13, A
GSB "DEC"
FXD ; PRT "COUNT = ", R2-1
END
```

4. Resetting: Clears toggled bits and prepares the card for generating a new interrupt.

5. Disarming: Terminates the transfer operation.

6-224 Initializing. Example 48 illustrates initializing the card by sending a data word with logical ones in bit positions BØØ through B11 (i. e., octal 7777). Notice that a control word containing SYE only is sent first. Although SYE has no effect on this card; it is included in the control word to avoid disabling the outputs of previously programmed cards.

6-225 The card may also be initialized by first reading a return data word from the card and then sending the same word back in the output mode. In this case, the output data will be a variable. Example 49 illustrates this technique.



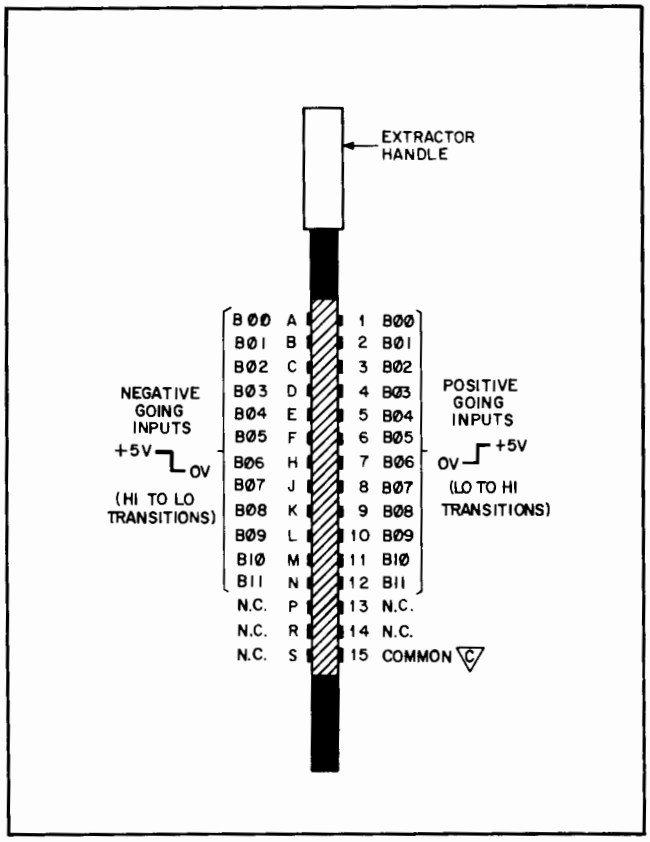
69436A Block Diagram

6-226 Arming and Reading the Card. Once the card is initialized, it must be armed before it can generate an interrupt and set the bus service request. The process interrupt card may be armed as follows:

1. Sending a control word with ISL on, followed by an address word containing the card address and a gate code "T", then another control woth with IEN and TME.

or

2. If the W6 jumper is installed, simply sending and



69436A Input Connector

Example 48. Initializing Using 7777_8:

9830A Calculator

```
11Ø CMD "?U7", "ØØ4ØTC7777T"
```

9820/21A Calculators

```
CMD "?U7", "ØØ4ØTC7777T"
```

Example 49. Initializing Using the Return Data Word:

9830A Calculator

```
11Ø CMD "?U7", "ØØ24ØTCX", "?5W"
12Ø ENTER (13, * ) A
13Ø CMD "?U7", "ØØ4ØT"
14Ø FORMAT F1ØØ5.Ø
15Ø OUTPUT (13,14Ø) "C"A"T";
```

9820/21A Calculators

```
CMD "?U7", "ØØ24ØTCX", "?5W"
FMT * ; RED 13, A
CMD "?U7", "ØØ4ØT"
FMT Z, "C", FXD * .Ø; WRT 13, A
FMT Z, "T", FXD * .Ø; WRT 13
```

gating a control word with IEN and TME on.

NOTE

If the W6 jumper is installed on a process interrupt card, it will allow the card to be armed whenever a control word with IEN is gated. This precludes selectively arming this card or any other card with W6 installed. This method may cause programming problems if used indiscriminately (see Paragraph 6-54).

6-227 Example 50 assumes that the W6 jumper has been removed, allowing the user to selectively arm cards. The second program lines shown in Example 50 test to determine whether the service request has been set, while the fourth lines test to determine if the multiprogrammer system is the bus device that requested service. The seventh lines examine the IRQ bit of the addressed card (in this case the 69436A card in slot 403 (C)). If the process interrupt card generated the service request, then the IRQ bit is subtracted from the return data word and the new value is stored in variable "A".

NOTE

Unlike the 69434A Event Sense Card, the Process Interrupt Card return data word is continually updated to reflect any bit transitions. If it is desired that the original interrupting word be retained in storage, or if comparison to a reference word is desired, an event sense card must be used as described in Paragraph 6-197.

6-228 Resetting the Card. Once the card has generated an interrupt and been read, it must either be reset and/or disarmed. Example 51 illustrates resetting a process interrupt card. The card is reset when the return data word (minus the IRQ bit) is sent back to the card in a manner similar to the method given in Paragraph 6-225 for initializing the card. Since the reset operation clears only those bits which were detected prior to the read operation, other bit transitions which occur during the read operation or afterwards are not lost, but are detected and stored, and may be sent to the calculator during the next input cycle.

NOTE

It is important that the process interrupt card not be read and reset if no interrupt has occurred, since the reset data word (all zeroes in this case) will disarm the card. After it is reset, the card is cleared to detect further bit transitions, in which case the system will be returned to the interrupt mode, or it may be disarmed.

6-229 Disarming the Card. It is not necessary, but would generally be considered good practice, to reset process interrupt cards before disarming them. Example 52 illustrates disarming a process interrupt card. Once disarmed, the cards cannot interrupt, but the edge detectors and storage registers are not disabled, thus allowing any further bit transitions to be detected and stored for possible future evaluation. Note that "CT" is the same as "C0000T".

6-230 Sample Test Program. The sample test program will enable the user to verify operation of 69436A cards.

Example 50. Arming and Reading a Process Interrupt Card:

9830A Calculator

```
160 CMD "?U7", "0024@TCT0046@T"
                .
                .
                .
200 IF STAT 13 > 1 THEN 500
210 GOSUB 2000
220 IF A4 # 64 THEN 500
230 CMD "?U7", "0024@TCX", "?5W"
240 ENTER (13, *) A
250 IF A < 10000 THEN 500
260 A = A - 10000
                .
                .
                .
500 REM CONTINUE PROGRAM
```

9820/21A Calculators

```
CMD "?U7", "0024@TCT0046@T"
                .
                .
                .
IF RDS 13 > 1; GTO "CONT"
GSB "SPOLL"
IF R4 ≠ 64; GTO "CONT"
CMD "?U7", "0024@TCX", "?5W"
FMT *; RED 13, A
IF A ≤ 9999; GTO "CONT"
A - 10000→A
                .
                .
                .
"CONT" CONTINUE PROGRAM
```

6-231 Test Procedure. Perform the following steps when using the 69436A sample test program.

1. Load the program into the calculator, being sure to include the appropriate subroutines.

2. Depressing RUN and EXECUTE on the 9830 calculator, or END, EXECUTE and RUN PROGRAM on the 9820/21 calculators will allow the program to run.

3. The program will halt with the GATE indicator on the 59500A lit. Momentarily connect a jumper between pin 15 and any pin 1 through 12 or the corresponding lettered pins A through N on the card edge connector. The calculator will wait indefinitely until a bit is toggled.

4. After a bit is toggled, the calculator will print (9820/21) or display (9830) the bit and pin number.

5. Toggling a new bit (or the same bit) and pressing CONT and EXECUTE on the 9830 calculator or RUN PROGRAM on the 9820/21 calculators will cause the calculator to display the new bit and pin number, (9830) or print the new bit and pin number (9820/21) and display "TEST COMPLETE".

6-232 Breadboard Input Card, 69480A

6-233 The 69480A breadboard input card is a simple multiprogrammer interface card, which allows the user to design, build, and control special input circuits through the multiprogrammer system. The following discussion will familiarize the user with programming the breadboard input card 69480A. All examples will assume the input card is installed in unit Ø, slot 412 (L).

6-234 Reading the Card. The return data word is read-back in octal form to variable "A". It is then up to the user to process the data. For example, the data could be used in octal form, or converted to decimal by the octal to decimal subroutine in Appendix A. In Example 53, the breadboard input card is read without a gate, as indicated by a gate code "X". This is necessary because there is no storage or CTF drive capability on this card. The only requirements for reading data from this card are:

1. A control word with ISL on must have been previously sent.

and

2. The card must be addressed.

6-235 Sample Test Program. This program will enable the user to verify operation of 69480A cards.

6-236 Test Procedure. Perform the following steps when using the 69480A sample test program.

1. Load the program into the calculator, being sure to include the appropriate subroutines.

2. On the input connector of the digital input card, temporarily install a jumper between pin 15 and any user selected pin between 1 and 12.

3. Depressing RUN and EXECUTE on the 9830

Example 51. Resetting a Process Interrupt Card:

9830A Calculator

```
270 CMD "?U7", "ØØØ4ØT"
280 FORMAT F1ØØ5.Ø
290 OUTPUT (13,28Ø) "C"A"T";
                .
                .
                .
CONTINUE PROGRAM
```

9820/21A Calculators

```
CMD "?U7", "ØØØ4ØT"
FMT Z, "C", FXD * .Ø; WRT 13, A
FMT Z, "T", FXD * .Ø; WRT 13
                .
                .
                .
CONTINUE PROGRAM
```

Example 52. Disarming a Process Interrupt Card:

9830A Calculator

```
3ØØ CMD "?U7", "ØØØ4ØTCT"
                .
                .
                .
5ØØ REM CONTINUE PROGRAM
```

9820/21A Calculators

```
CMD "?U7", "ØØØ4ØTCT"
                .
                .
                .
"CONT" CONTINUE PROGRAM
```

69436A Sample Test Program

9830A Calculator

```
10 GOSUB
20 C = 0
30 CMD "?U7", "00040TC7777T00240TCT00460T"
40 IF STAT 13 > 1 THEN 40
50 GOSUB 2000
60 CMD "?U7", "00240TCX"
70 CMD "?5W"
80 ENTER (13, * ) A
90 B = A = A – 10000
100 GOSUB 2200
110 GOSUB 2600
120 CMD "?U7", "00040T"
130 FORMAT F1005.0
140 OUTPUT (13,130) "C"B"T";
150 CMD "?U7", "00040TCT"
160 IF C = 1 THEN 210
170 STOP
180 C = 1
190 CMD "?U7", "00240TCT00460T"
200 GOTO 40
210 END
```

9820/21A Calculators

```
GSB "SPOLL"
0→C
CMD "?U7", "00040TC7777T00240TCT00460T"
IF RDS 13 > 1; JMP 0
GSB "SPOLL"
CMD "?U7", "00240TCX"
CMD "?5W"
FMT * ; RED 13, A
A – 10000→A→B
GSB "DEC"
GSB "BIT"
CMD "?U7", "00040T"
FMT Z, "C", FXD * .0 ; WRT 13, B
FMT Z, "T", FXD * .0; WRT 13
CMD "?U7", "00040TCT"
IF C = 1, GTO "END"
STP
1→C
CMD "?U7", "00240TCT00460T"
JMP –17
"END"; DSP "TEST COMPLETE"
END
```

Example 53. Reading a Breadboard Input Card:

9830A Calculator

```
110 CMD "?U7", "00240TLX", "?5W"
120 ENTER (13, * ) A
      .
      .
CONTINUE PROGRAM
```

9820/21A Calculators

```
CMD "?U7", "00240TLX", "?5W"
FMT * ; RED 13, A
      .
      .
CONTINUE PROGRAM
```

69480A Sample Test Program

9830A Calculator

```
10 CMD "?U7", "00240TLX", "?5W"
20 ENTER (13, * ) A
30 A = 7777 – A
40 GOSUB 2200
50 GOSUB 2600
60 END
```

9820/21A Calculators

```
CMD "?U7", "00240TLX", "?5W"
FMT * ; RED 13, A
7777 – A→A
GSB "DEC"
GSB "BIT"
END
```

calculator or END, EXECUTE and RUN PROGRAM on 9820/21 calculators will allow the program to run. The calculator will stop with the input bit and pin number displayed (or printed). If no data line (pins 1-12) has been jumpered to pin 15, the calculator will stop with "NO DATA RECEIVED" displayed.

4. Repeating steps 2 and 3 will enable the user to check each data input line on the card.

6-237 Resistance Output Cards 69500A – 69513A

6-238 These cards provide a programmed value of resistance as their output. Twelve magnetically shielded, mercury-wetted, reed relays select the resistance values by modifying the value of a series string of high-accuracy, binary weighted resistors.

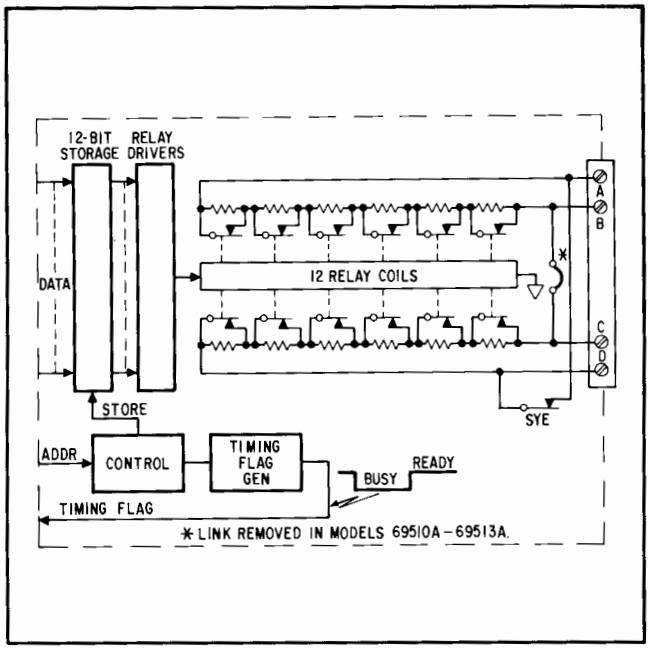
1. Model 69500A is supplied without output resistors so that customers may select and load their own resistors if desired.

2. Models 69501A – 69506A are single output, 12-bit resolution cards designed to program the voltage output of a single HP power supply equipped with Option 040.

3. Models 69510A – 69513A are dual output, 6-bit resolution cards designed to program the current outputs of two HP power supplies equipped with Option 040.

6-239 The following discussion will familiarize the user with programming the resistance output cards 69500A – 69513A. All examples will assume the card is installed in unit Ø, slot 4Ø3 (C). For 69510A – 69513A cards, channel 1 refers to the output programmed by the lower order bits, BØØ – BØ5, and channel 2 refers to the output programmed by the higher order bits, BØ6 – B11. Channel 1 resistance output terminals are C and D while channel 2 resistance output terminals are A and B.

6-240 Resistance Outputs. Octal data transmitted to program a resistance card will program one 12-bit resistance output from 69500A – 69506A resistance cards, or two 6-bit resistance outputs from 69500A – 69506A resistance cards. When programming a 69510A – 69513A card, the two least significant digits in the data field will program channel 1 (BØØ –BØ5), and the two most significant digits will program channel 2 (BØ5 – B11). Table 6-7 lists the maximum resistance outputs and resolution of each card.



69500A – 69513A Block Diagram

6-241 The following steps will yield the data value necessary to program a resistance output card to the desired resistance. For 69500 – 69506 cards, or single channel (channel 1) programming of 69510 – 69513 cards, it is only necessary to perform steps 1 and 2. For dual channel programming of 69510A – 69513A cards, perform steps 1 through 4.

1. Divide the required output resistance in ohms by the resolution of the card for each channel.

2. Calculate the octal equivalent of the value yielded in step 1.

3. Multiply the octal value obtained for channel 2 by 100 (decimal).

4. Add the values obtained for channels 1 and 2 together.

For example, to calculate the octal data value required to program 4128 ohms from a 69501A resistance card;

1. 4128/2 = 2064

2. Octal equivalent of 2064 = 4020

Or, to calculate the octal data value required to program a 69510A card to 64 ohms on channel 1 and 128 ohms on channel 2;

1. 64/4 = 16

2. Octal equivalent of 16 = 20 (channel 1)

3. 128/4 = 32

4. Octal equivalent of 32 = 40 (channel 2)

5. 40 x 100 = 4000

6. 20 + 4000 = 4020

6-242 Programming Examples. Examples 54 and 55 illustrate programming resistance from a resistance output card. A control word is sent first containing SYE only, followed by a data word containing the card slot address, and the output data desired.

6-243 Although a resistance output card returns a CTF flag as a function of internal timing circuits, TME is not included in the control words of the programming examples. This is because the statement execution time of 9820/21 and 9830 calculators is greater than the period of the internal timing circuits (6 milliseconds). If the users system requires each card to complete its operation before programming the next card, the user has the choice of only programming one card per program statement, or using TME in the control words. If TME is used, the multiprogrammer system will set its service request when the card(s) finish processing the output data. In this case, a serial poll of the multiprogrammer system must be done to reset service request. If the user has modified the flag circuits on the card to lengthen the CTF flag, as described in the card manual, it will be necessary to use TME and consequently, a serial poll routine.

6-244 Example 54 illustrates using a data constant of $4Ø2Ø_8$ to program a resistance output card located in slot 4Ø3 (C).

6-245 Example 55 illustrates using a data variable ("A2" or "R2") to program a resistance output card.

6-246 Programming Power Supplies Equipped With Option 040. Up to this point the discussion has focused on programming a resistance output from a resistance output card. However, it is not necessary to calculate the resistance to program an Option 040 power supply to a desired voltage or current. Instead, selection of a resistance card corresponding to a specific Option 040 power supply changes the calculations to the following steps.

1. Divide the required output voltage or current by the LSB (see Paragraph 6-247) of the power supply/resistance card combination.

2. Convert the decimal value to octal.

3. If a dual channel resistance card is being used, multiply the octal value obtained for channel two by 100, then add channels one and two together, as shown previously when programming resistance outputs.

Table 6-7. Maximum Output Resistance and Resolution

Model	Maximum Resistance (Ohms)	Resistance (Ohms)
69500A *		
69501A	8,190	2
69502A	30,712.5	7.5
69503A	61,425	15
69504A	40,950	10
69505A	81,900	20
69506A	204,750	50
69510A **	252	4
69511A **	945	15
69512A **	1,260	20
69513A **	1,890	30

* Resistance values on the 69500A are determined by the user.

** Maximum resistance and resolution are specified per channel.

6-247 The LSB of the power supply/resistance card combination may be determined as follows.

1. For resistance cards 69500A – 69506A, the LSB is .01 volts for power supplies up to 40 volts, and .025 volts for power supplies 41-100 volts.

2. For resistance cards 69510A – 69513A, the LSB is equal to 2% of the maximum rated current of the power supply, with the exception of the 6105A, where the LSB is equal to 1.875% of the maximum rated current.

6-248 Examples 56 and 57 are subroutines which will allow the user to program Option 040 power supplies to any voltage or current desired, within the capability of the supply. Since it is not possible to program voltage or current values with greater accuracy than the LSB of the power supply/resistance card combination, any voltage (or current) value which is not a multiple of the LSB will be rounded off to the nearest value which is a multiple of the LSB.

6-249 Example 56 is a subroutine which allow the user to program a voltage output from an Option 040 power supply with a maximum rating of 10 volts. The subroutine must be entered with a variable "A", equal to the output voltage desired. If an attempt is made to program either a negative voltage, or a voltage exceeding the maximum rated voltage of the power supply, the calculator will stop the program and display VOLTAGE ERROR. To continue the program, the user must depress CONT and EXECUTE on a 9830A calculator, or RUN PROGRAM on 9820/21A calculators. In this case, the program will be continued but the subroutine will not process the requested input.

6-250 In order to use the subroutine in Example 56 for programming power supplies with outputs other than 10 volts, the following changes must be made.

1. Change the maximum limit in the second program line to reflect the maximum rated output voltage of the power supply. For example, if it is desired to program a

40 volt power supply, change the second program line to read as follows:

9830A: 121Ø IF A > – .ØØ1 AND A < 4Ø + .ØØ1 THEN 1250

9820/21A: AF (A > – .ØØ1) (A ⩽ 4Ø) ; JMP 4

2. If the maximum rated output of the power supply is greater than 40 volts, change the sixth program line to read as follows:

9830A: 125Ø A = A/.Ø25

9820/21A: A/.Ø25 A

6-251 Example 57 is a subroutine which allows the user to program current outputs from two Option 040 power supplies with a maximum rating of 1 amp each. The subroutine must be entered with two variables, "B" and "C", equal to the output current, in amps, desired from each supply. Variable "B" specifies the desired output current from the power supply connected to channel 1, while variable "C" specifies the desired output current from the power supply connected to channel 2.

6-252 If an attempt is made to program either a negative current, or a current exceeding the maximum rated current of either power supply, the calculator will stop the program and display "I-ERROR,CH1" or "I-ERROR,CH2", corresponding to the channel that has been overprogrammed. To continue the program after a stop occurs, depress CONT and EXECUTE on a 9830A calculator, or RUN PROGRAM

Example 54. Programming Resistance Output Using Data Constants:

9830A Calculator

```
11Ø CMD "?U7", "ØØØ4ØTC4Ø2ØT"
              .
              .
CONTINUE PROGRAM
```

9820/21A Calculators

```
CMD "?U7", "ØØØ4ØTC4Ø2ØT"
              .
              .
CONTINUE PROGRAM
```

Example 55. Programming Resistance Output Using Data Variables:

9830A Calculator

```
11Ø CMD "?U7", "ØØØ4ØT"
12Ø FORMAT F1ØØ5.Ø
13Ø OUTPUT (13,12Ø) "C"A2"T";
              .
              .
CONTINUE PROGRAM
```

9820/21A Calculators

```
CMD "?U7", "ØØØ4ØT"
FMT Z, "C", FXD * .Ø; WRT 13, R2
FMT Z, "T", FXD * .Ø; WRT 13
              .
              .
CONTINUE PROGRAM
```

Example 56. Programming Voltage from an Option 040 Power Supply:

9830A Calculator

```
12ØØ REM OPT Ø4Ø VOLTAGE OUTPUT
121Ø IF A > – .ØØ1 AND A < 1Ø + .ØØ1 THEN 125Ø
122Ø DISP "VOLTAGE ERROR"
123Ø STOP
124Ø GO TO 13ØØ
125Ø A = A/.Ø1
126Ø GOSUB 24ØØ
127Ø CMD "?U7", "ØØ14ØT"
128Ø FORMAT F1ØØ5.Ø
129Ø OUTPUT (13,128Ø) "C"A2"T";
13ØØ RETURN
```

9820/21A Calculators

```
"VØ4Ø"
IF (A > – .ØØ1) (A ⩽ 1Ø) ; JMP 4
DSP "VOLTAGE ERROR"
STP
JMP 6
A/.Ø1 → A
GSB "OCT"
CMD "?U7", "ØØ14ØT"
FMT Z, "C", FXD * .Ø; WRT 13, R2
FMT Z, "T", FXD * .Ø; WRT 13
RET
```

on 9820/21A calculators. In this case, the program will be continued, but the subroutine will not process the requested current(s). However, if channel 1 has been properly programmed and channel 2 has been overprogrammed, variable "B", the entry variable for channel 1, will now represent the octal equivalent of the desired current. In this case, when variable "C" is corrected, variable "B" must also be reset to the desired current value before re-entering the subroutine.

6-253 In order to use the subroutine in Example 57 to program a single power supply, or power supplies with outputs other than one ampere, the following changes must be made.

1. To program the current from a single power supply, delete lines 147Ø - 154Ø from the 9830 program, or the eighth through the fifteenth lines of the 9820/21 program. Variable "C" becomes unnecessary, and the output will be from channel 1.

NOTE

If user selection of a single output channel is desired, the subroutine may be used as is. In this case, the variable corresponding to the channel not being programmed must be set to equal zero.

2. For power supplies with outputs other than one ampere, the second and ninth program line must be changed so the maximum current outputs of the power supplies are multiplied by .02. For example, if it is desired to program a 1.5 amp power supply, change the second and ninth program line as follows:

9830A: 141Ø A = B/(1.5 * .Ø2)
148Ø A = C/(1.5 * .Ø2)

9820/21A: (2) B/(1.5 * .Ø2)→A
(9) C/(1.5 * .Ø2)→A

NOTE

If a 6105A power supply is being programmed, the multiplication factor becomes .01875. This allows 53 steps to be programmed, instead of the 50 steps available with other supplies. Therefore, the step-limit value of 51 (9830) or 50 (9820/21) should be increased to 55 (9830) in lines 142Ø and 149Ø and 54 (9820/21) in the third and tenth program lines.

6-254 69500A – 69506A Sample Test Program. This program will enable the user to verify operation of resistance output cards 69500A – 69506A. The sample program allows the user to program a resistance value from a 69501A card. To program 69500A, 69502A – 69506A cards, line 4Ø of the 9830 program or the third line of the 9820/21A program must be changed so variable "A" is divided by the resolution of the card as given in Table 6-7.

Example 57. Programming Current from an Option 040 Power Supply:

9830A Calculator

```
14ØØ REM OPT Ø4Ø CURRENT OUTPUT
141Ø A = B/(1 * .Ø2)
142Ø IF A > –.ØØ1 AND A < 51 THEN 146Ø
143Ø DISP "I-ERROR, CH1"
144Ø STOP
145Ø GO TO 158Ø
146Ø GOSUB 24ØØ
147Ø B = A2
148Ø A = C/(1 * .Ø2)
149Ø IF A > – .ØØ1 AND A < 51 THEN 153Ø
15ØØ DISP "I-ERROR, CH2"
151Ø STOP
152Ø GO TO 158Ø
153Ø GOSUB 24ØØ
154Ø A2 = A2 * 1ØØ + B
155Ø CMD "?U7", "ØØ14ØT"
156Ø FORMAT F1ØØ5.Ø
157Ø OUTPUT (13,156Ø) "C"A2"T";
158Ø RETURN
```

9820/21A Calculators

```
"IØ4Ø"
B/(1 * .Ø2)→A
IF (A > –.ØØ1) (A ≤ 50; JMP 4
DSP "I-ERROR, CH1"
STP
JMP 13
GSB "OCT"
R2→B
C/ (1 * .Ø2) – A
IF (A > –.ØØ1) (A ≤ 5Ø) ; JMP 4
DSP "I-ERROR, CH2"
STP
JMP 6
GSB "OCT"
R2 * 1ØØ + B→R2
CMD "?U7", "ØØ14ØT"
FMT Z, "C", FXD * .Ø; WRT 13, R2
FMT Z, "T", FXD * .Ø; WRT 13
RET
```

NOTE

Since resistance values programmed must always be a multiple of the resolution (LSB) of the card, any value which is not a multiple of the card LSB will be rounded off to the nearest value which is a multiple of the LSB.

6-255 69500A – 69506A Test Procedure. Perform the following steps when using the 69500A – 69506A sample test program.

1. Load the program into the calculator, being sure to include the appropraite subroutines.

2. Depressing RUN and EXECUTE on the 9830 calculator, or END, EXECUTE, and RUN PROGRAM on 9820/21 calculators will allow the program to run.

3. The calculator will stop with ENTER RESISTANCE displayed. The user must enter a resistance value, no greater than the maximum capability of the card (as listed in Table 6-7). Then depress EXECUTE on the 9830 calculator or RUN PROGRAM on 9820/21 calculators.

4. After running the program, the calculator will display "TEST RESISTANCE". At this point, the user should use an ohmmeter to measure the resistance output between terminals C and D on the card. Note that a 10 ohm zero calibration resistor is in series with the output resistance of the 69501A – 69513A cards.

6-256 69510A – 69513A Sample Test Program. This program will enable the user to verify operation of resistance output cards 69510A – 69513A. The program allows the user to program separate resistance values from channel 1 and channel 2 of a 69510A card. To program 69511A – 69513A cards, lines 40 and 90 of the 9830 program or the third and seventh lines of the 9820/21 program must be changed so that variable "A" is divided by the resolution of the card as given in Table 6-7.

6-257 69510A – 69513A Test Procedure. Perform the following steps when using the 69510A – 69513A sample test program.

1. Load the program into the calculator, being sure to include the appropriate subroutines.

2. Depressing RUN and EXECUTE on the 9830 calculator, or END, EXECUTE, and RUN PROGRAM on 9820/21 calculators will allow the program to run.

3. The calculator will stop with "ENTER R1" displayed. Ths user must enter a resistance value for channel 1, no greater than the maximum capability of the card (as listed in Table 6-7), then depress EXECUTE on the 9830 calculator or RUN PROGRAM on 9820/21 calculators.

4. The calculator will stop with ENTER R2 displayed. The user must enter a resistance value for channel 2, no greater than the maximum capability of the card (as listed in Table 6-7), then depress EXECUTE on the 9830 calculator or RUN PROGRAM on 9820/21 calculators.

5. After running the program, the calculator will display "TEST RESISTANCE". At this point, the user should use an ohmmeter to measure the resistance output as follows.

a. The resistance output of channel 1 may be measured between terminals C and D.

b. The resistance output of channel 2 may be measured between terminals A and B.

6-258 Programmable Timer Card, 69600A

6-259 This card generates a crystal-controlled one-shot pulse each time it is commanded by the program. The duration of the pulse is determined by the combination of two factors: (1) the number of programmed time increments,

69500A – 69506A Sample Test Program

9830A Calculator

```
10 GOSUB 2000
20 DISP "ENTER RESISTANCE";
30 INPUT A
40 A = A/2
50 GOSUB 2400
60 CMD "?U7", "00040T"
70 FORMAT F1005.0
80 OUTPUT (13,70) "C"A2"T";
90 DISP "TEST RESISTANCE"
100 END
```

9820/21A Calculators

```
GSB "SPOLL"
ENT "ENTER RESISTANCE", A
A/2→A
GSB "OCT"
CMD "?U7", "00040T"
FMT Z, "C", FXD * .0 ; WRT 13, R2
FMT Z, "T", FXD * .0; WRT 13
DSP "TEST RESISTANCE"
END
```

from 1 to 4095; and (2) the selected period of the increments, from 1 microsecond to .1 second. Period selection can be accomplished by fixed jumpers on the 69600A card or brought under program control by using a 69330A relay output card. The following discussion will familiarize the user with programming timer card 69600A. All examples will assume the timer card is installed in unit Ø, slot 414 (N).

6-260 Operating Modes. Basically there are three operating modes which apply to this card.

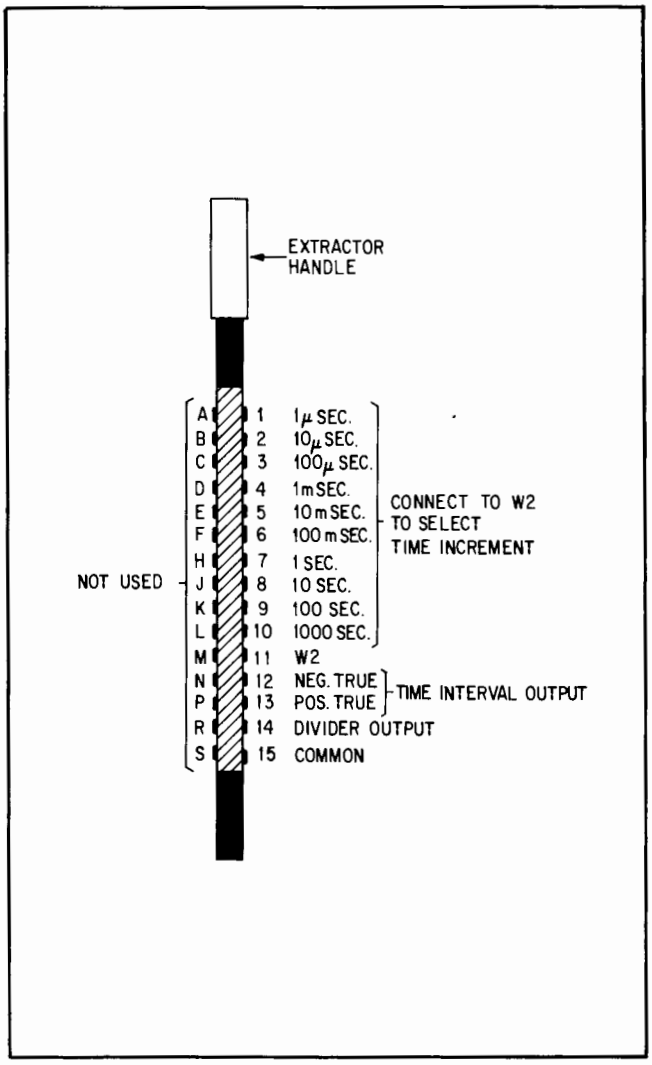
1. Automatic Handshake Mode: Allows the user to program an output pulse of a specified duration, but does not set service request upon completion of the pulse.

2. Timing Mode: Allows the user to program an output pulse of a specified duration from a single card and sets service request upon completion of the pulse. Jumper W3 must be installed on 69600A card for this mode of operation. With W3 installed, CTF is held busy for the duration of the output pulse.

3. Interrupt Mode: Allows the user to program an output pulse of specified duration from multiple cards and sets service request upon completion of the pulse. Jumper W3 is omitted on 69600A for this mode of operation.

NOTE

Throughout the following discussion, the user should keep in mind that the number of time increments is programmable, but the actual period of the increments is selected by installing appropriate jumpers on the card.



69600A Output Connector

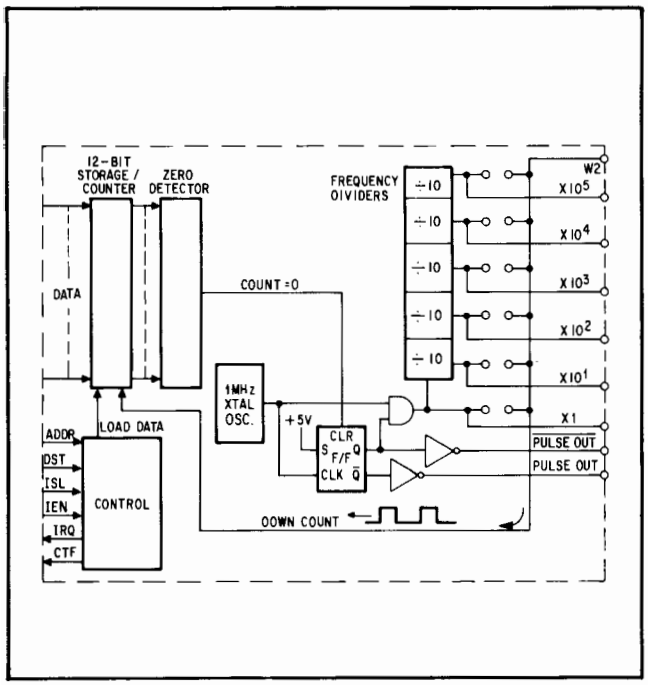
69510A – 69513A Sample Test Program

9830A Calculator

```
1Ø GOSUB 2ØØØ
2Ø DISP "ENTER R1";
3Ø INPUT A
4Ø A = A/4
5Ø GOSUB 24ØØ
6Ø B = A2
7Ø DISP "ENTER R2";
8Ø INPUT A
9Ø A = A/4
1ØØ GOSUB 24ØØ
11Ø C = A2 * 1ØØ
12Ø A2 = B + C
13Ø CMD "?U7", "ØØØ4ØT"
14Ø FORMAT F1ØØ5.Ø
15Ø OUTPUT (13,14Ø) "C"A2"T";
16Ø DISP "TEST RESISTANCE"
17Ø END
```

9820/21A Calculators

```
GSB "SPOLL"
ENT "ENTER R1", A
A/4→A
GSB "OCT"
R2→B
ENT "ENTER R2", A
A/4→A
GSB "OCT"
R2 * 1ØØ→C
B + C→R2
CMD "?U7", "ØØØ4ØT"
FMT Z, "C", FXD * .Ø; WRT 13, R2
FMT Z, "T", FXD * .Ø; WRT 13
DSP "TEST RESISTANCE"
END
```



69600A Block Diagram

6-261 Automatic Handshake Mode. This mode requires sending a control word with SYE only followed by a data word containing the card slot address and the number of time increments desired (in octal). Example 58 illustrates using a data constant of 144_8 to specify the number of time increments (duration of output pulse).

6-262 In some cases it might be desirable to use a variable to specify the number of time increments desired. The value could then be changed as a function of the user's program. Example 59 illustrates the format required to output variables. In this example, "A2" (or "R2" when using 9820/21 calculators) is the variable which specifies the number of timer intervals desired. Again, it must be kept in mind that this must be an octal value.

6-263 Timing Mode. Example 60 illustrates the timing mode of operation. To utilize this mode the user must first install jumper W3 on the timer card. Notice that TME is included in the control word and the bus service request is used to indicate completion of the pulse. Example 60 is written in the format normally used to output variable time increments. Of course it could easily be written to output fixed octal values by eliminating the second and third program lines and rewriting the first line as follows:

CMD "?U7", "0Ø04ØTN7777T00160T"

6-264 In the rewritten first line, 7777 is a fixed octal value representing the number of time intervals desired. In either case (fixed or variable), when the timer card pulse is complete the bus service request line will be set.

6-265 Interrupt Mode. Example 61 illustrates the interrupt mode of operation. Programming the timer card in the interrupt mode consists of four basic steps:

1. The timer card must first be loaded by sending a control word to the multiprogrammer with ISL off, followed by a data word containing the slot address of the card and the number of time increments desired. The data word is transmitted when a gate, as indicated by a gate code "T". This is necessary to load the data into the card.

2. The timer card must now be armed as follows:

a. Sending a control word with ISL on, followed by an output word containing the card address and a gate code "T", then another control word with IEN and TME.

or

b. If W6 jumper is installed, simply sending and gating a control word with IEN and TME on. This precludes selectively arming this card or any other card with W6 installed. Arming with IEN (W6 installed) may cause programming problems if used indiscrimenately (see paragraph 6-54).

3. When an interrupt occurs (programmed pulse is completed), the bus service request line will be set. A poll of the previously return armed cards must then be performed by addressing the cards in the input mode (ISL on), reading the card (without a gate) and examining the state of IRQ. If the return data word is equal to or greater than 10000 it indicates the card had interrupted and the pulse is complete.

4. The card should now be disarmed by addressing the card with ISL off, zero data, and a gate.

6-266 Example 61 assumes that jumper W6 is removed allowing the user to selectively arm the cards. This example uses data variables to specify the number of time increments desired. Example 61 can be changed to output data constants (e.g. 7777_8) by eliminating the second and third program lines and rewriting the first line as follows:

W6 out: CMD "?U7","0Ø04ØTN7777T0Ø24ØTNT0Ø46ØT"
or
W6 in: CMD "?U7","0Ø04ØTN7777T0Ø46ØT"

6-267 Sample Test Program. This program will enable the user to verify operation of 69600A cards.

6-268 Test Procedure. Perform the following steps when using the 69600A sample test program.

1. Load the program into the calculator, being sure to include the appropriate subroutines.

2. On the output connector of the timer card, temporarily install a jumper between pin 6 and pin 11.

Example 58. Programming a Pulse in the Automatic Handshake Mode Using Data Constants:

9830A Calculator

```
110 CMD "?U7",",0004ØTN144T"
                .
                .
                .
CONTINUE PROGRAM
```

9820/21A Calculators

```
CMD "?U7",",0004ØTN144T"
                .
                .
                .
CONTINUE PROGRAM
```

Example 59. Programming a Pulse in the Automatic Handshake Mode Using Data Variables:

9830A Calculator

```
110 CMD "?U7",",0004ØT"
120 FORMAT F1005.Ø
130 OUTPUT (13,12Ø)"N"A2"T";
                .
                .
                .
CONTINUE PROGRAM
```

9820/21A Calculators

```
CMD "?U7","0004ØT"
FMT Z,"N",FXD *.Ø;WRT 13,R2
FMT Z,"T",FXD *.Ø;WRT 13
                .
                .
                .
CONTINUE PROGRAM
```

Example 60. Timing Mode:

9830A Calculator

```
110 CMD "?U7",",0004ØT"
120 FORMAT F1005.Ø
130 OUTPUT (13,12Ø)"N"A2"T00160T";
                .
                .
                .
200 IF STAT 13 > 1 then 500
210 GOSUB 2000
220 IF A4#64 THEN 500
230 REM PULSE IS COMPLETE
                .
                .
                .
500 REM CONTINUE PROGRAM
```

9820/21A Calculators

```
CMD "?U7",",0004ØT"
FMT Z,"N",FXD *.Ø;WRT 13,R2
FMT Z,"T00160T,FXD *.Ø;WRT 13
                .
                .
                .
IF RDS 13 > 1;GTO"CONT"
GSB "SPOLL"
IF R4≠64;GTO "CONT"
  (PULSE IS COMPLETE)
                .
                .
                .
"CONT" CONTINUE PROGRAM
```

3. If desired, the user may connect a voltmeter between pins 13 and 15 to observe the output pulse. In this case pin 15 is common.

4. Depressing RUN and EXECUTE on a 9830 calculator or END, EXECUTE, and RUN PROGRAM on 9820/21 calculators will cause the calculator display to go blank for approximately 10 seconds, then display "TEST COMPLETE". If a voltmeter was connected in step 3, the output pulse will go positive for approximately 10 seconds, corresponding to the blank calculator display.

Example 61. Interrupt Mode:

9830 Calculator

```
110 CMD "?U7",",00040T"
120 FORMAT F1005.0
130 OUTPUT (13, 120)"N"A2"T00240TNT00460T";
                    .
                    .
                    .
200 IF STAT 13 > 1 THEN 500
210 GOSUB 2000
220 IF A4#64 THEN 500
                    .
                    .
                    .
230 CMD "?U7",",00240TNX",",?5W"
240 ENTER (13,*)A
250 IF A< 10000 THEN 500
260 REM PULSE IS COMPLETE
270 CMD "?U7", "00040TNT"
                    .
                    .
                    .
500 REM CONTINUE PROGRAM
```

9820/21A Calculators

```
CMD "?U7",",00040T"
FMT Z,"N",FXD *.0;WRITE 13,R2
FMT Z,"T00240TNT00460T",FXD *.0;WRT 13
                    .
                    .
                    .
IF RDS 13 > 1;GOTO "CONT"
GSB "SPOLL"
IF R4≠64;GTO "CONT"
                    .
                    .
                    .
CMD "?U7",",00240TNX",",?5W"
FMT *;RED 13,A
IF A≤9999;GTO "CONT"
  (PULSE IS COMPLETE)
CMD "?U7",",00040TNT"
                    .
                    .
                    .
"CONT" CONTINUE PROGRAM
```

69600A Sample Test Program

9830 Calculator

```
10 GOSUB 2000
20 CMD "?U7", "00040TN144T00240TNT00460T"
30 B=STAT13
40 IF B> 1 THEN 30
50 GOSUB 2000
60 IF A4#64 THEN 30
70 CMD "?U7",",00040TNT"
80 DISP "TEST COMPLETE"
90 END
```

9820/21 Calculators

```
GSB "SPOLL"
CMD "?U7",",00040TN144T00240TNT00460T"
RDS 13→B
IF B>1;JMP -1
GSB "SPOLL"
IF R4≠64;JMP -3
CMD "?U7",",00040TNT"
DSP "TEST COMPLETE"
END
```

6-269 Frequency Reference Card, 69601A

6-270 This card is not programmable but can be used in conjunction with other plug-in cards. For example, the 69601A card can be used with pulse counter card 69435A for making time interval measurements (see paragraph 6-284). In addition, the frequency reference card can be used as a divider for TTL signals through use of the auxiliary oscillator input.

6-271 The frequency reference card provides six square-wave outputs at fixed frequencies of 1Hz, 10Hz, 100Hz, 1kHz, 10kHz, and 100kHz. The output frequencies are derived from an internal 1MHz crystal oscillator and can be turned off by the low state of an external TTL logic gate or external contact closure between the inhibit and common pins on the card.

6-272 MULTIPLE CARD PROGRAMS

6-273 The programs and explanations in this section provide examples of typical measurement applications and methods using several types of multiprogrammer cards in together. The techniques shown in this section are merely examples of how to solve certain measurement problems and should not be considered as an exhaustive

treatment of the subject. The user is encouraged to expand or change the methods shown here, to suit his particular requirements. The examples given below ((1) voltage scanning, (2) time interval measurement, and (3), frequency measurement) may be used as stand along programs or modified and incorporated as subroutines in a larger program. Any subroutines called from these examples (e.g., GOSUB 2000) are included in Appendix A.

6-274 Voltage Scanning and Measurement Using the 69421A and 69330A Cards

6-275 The following program illustrates using a 69421A voltage monitor card in conjunction with a 69330A relay card to measure six isolated voltages, store, the measured values, and then print the values (the 9830A calculator must be equipped with a 9866A printer). If the voltages to be measured share a common reference, it is possible to use this card combination to measure twelve different voltages. Additional 69330A cards and appropriate modifications to the program may be used to expand the scanner array if desired.

6-276 Example 62, assumes that the relay card is installed in slot 401 (A) and the voltage monitor card is installed in slot 402 (B). It is further assumed that the voltages to be measured are in the range of -10.240 through +10.235 volts, permitting use of a standard voltage monitor card. (Option 102 for the 69421A card may be used to measure voltages within the range of -102.40 through +102.35 volts). The voltage measurement program shown in paragraphs 6-161 through 6-170 is incorporated in this example as a subroutine (i.e., GOSUB 1000).

6-277 Numbered edge connector pins from the relay card are connected in pairs across the voltages being measured (i.e. 1-2, 3-4, etc.) with the even numbered pin connected to the positive side of the voltage to be sampled. Relay card pins A,C,E,H,K, and M are jumpered together and connected to pin R of the voltage monitor card and pins B,D,E,J, L, and N are connected to pin L of the voltage monitor card. In addition, relay card pins 14 and 13 are connected together and 15 is jumpered to P in order to complete the gate/flag handshake circuit.

Example 62. Voltage Measurement Using 69421A and 69330A Cards:

9830 Calculator Program

```
10 GOSUB 2000
20 FOR B=1 TO 6
30 READ B6
40 CMD "?U7","00140T"
50 FORMAT F1005.0
60 OUTPUT (13,50) "A"B6"T";
70 GOSUB 1000
80 B[B] = A
90 NEXT B
100 FOR B=1 to 6
110 PRINT "VOLTAGE ="B [B]
120 NEXT B
130 DATA 3,14,60,300,1400,6000
140 RESTORE
150 END

1000 REM MEASURE VOLTAGE
1010 CMD "?U7","00240TBT"
1020 WAIT 6
1030 CMD "?U7","BX" ,"?5W"
1040 ENTER (13,*)A
1050 GOSUB 2200
1060 IF A2 < = 2047 THEN 1080
1070 A2 = A2-4096
1080 A=.005*A2
1090 RETURN
```

9820/21A Calculator Program

```
GSB "SPOLL"
3→R7;14→R8;60→R9;300→R10;1400→R11;6000→R12
7→B
CMD "?U7" ,"00140T"
FMT Z,"A",FXD *.0;WRT 13,RB
FMT Z,"T",FXD *.0;WRT 13
GSB "MEAS"
A→RB
IF B≤ 11;B+1→B;JMP -5
7→B
FXD 3;PRT "VOLTAGE =",RB
IF B≤11;B+1→B;JMP -1
END

"MEAS";CMD "?U7","00240TBT"
"WAIT"
CMD "?U7","BX","?5W"
FMT *;RED 13,A
GSB "DEC"
IF R2≤2047;JMP 2
R2-4096→R2
.005*R2→A
RET
```

Explanation:

1Ø Check and clear service request

2Ø Establishes a "FOR. . . NEXT" loop with line 9Ø in order to increment a pointer to store the six voltage readings into an array "B" (line 8Ø). Lines 3,8, and 9 together with register R7 through R12 serve the same purpose in the 9820A/21A program.

3Ø Reads an octal value "B6" from the data array in line 13Ø and then increments the array pointer. This value is output to the multiprogrammer system in line 6Ø and programs the appropriate relay closures for each of the six voltage measurements. The initial data values in registers R7 through R12 perform the same function in the 9820A/21A program.

4Ø Establishes the output mode with DTE and SYE on.

5Ø-6Ø Formats and outputs data to program the desired contact closures on the 69330A card for each of the six voltage measurements.

7Ø Calls the voltage measurement subroutine (see paragraphs 6-161 through 6-170 for an explanation) and returns to the main program with a decimal value in volts.

8Ø Stores the voltage data returned in line 7Ø into an array (registers R7 through R12 in the 9820A/21A program)

9Ø Loops program until six voltage measurements are made.

1ØØ Establishes a new six step "FOR . . . NEXT" loop with line 12Ø.

11Ø Prints measured voltage

12Ø Increments array pointer to access the next stored voltage measurement and returns program to line 1ØØ until all six values are printed.

13Ø Data array containing octal values corresponding to the 69330A contact closures programmed in line 6Ø. Lines 2 and 5 of the 9820A/21A program contain the equivalent statements.

14Ø Returns the pointer in the line 13Ø data array to the first register. Line 2 of the 9820A/21A performs the same initialization function when first encountered during program execution.

NOTE:

More complete explanations of the specialized programming commands used in example 62 may be found in the operating and programming manuals for the particular calculator.

6-278 Frequency Measurement Using the 69435A and 69600A Cards

6-279 The following program illustrates using a 69435A pulse counter card and a 69600A programmable timer card to measure the frequency of a pulse train, and then displaying the frequency on the 9830A calculator or printing the frequency on the 9820A/21A calculators.

6-280 Example 63 assumes the pulse counter card is installed in slot 4Ø1 (A) and the timer card is installed in slot 4Ø2 (B). The pulse train frequency being measured should not exceed 200kHz.

6-281 A primary consideration when measuring frequency with the technique described here is the capacity of the pulse counter card. The timer card output is connected to the count enable input of the counter card and is programmed to select the gate time for the frequency measurement. The user must select gate times so that the maximum count count capacity of 4095 is not exceeded during the measurement period. This particular program automatically selects gate times as follows:

1. The initial gate time is 10 milliseconds, which produces 2000 counts at the maximum input frequency of 200kHz.

2. Whenever the total count is less than 100, the time interval is increased by a factor of ten and a new measurement is made. The maximum gate time in this program is one second.

6-282 The timer card will be used in the timing mode, requiring that the W3 jumper be installed on the card. Instead of testing for a service request as indication that the gate pulse is completed, however, the program transmits two control words in one line; the first establishes the timing mode and the second turns TME off and ISL on in order to read data from the counter card. This technique will cause the calculator to pause while the timer card

outputs the gate pulse, and then proceed to read the card and display (or print) the frequency.

6-283 Figure 6-1 shows the connections appropriate for this particular example. Timer card pins 11 (W2) and 4 should be jumpered together and pin 13 should be connected to pulse counter card pin A. Pulse counter card pins M and 15 should be jumpered together. The pulse train to be measured should be applied between the appropriate count-up input (as explained in the pulse counter card manual) and pin F (common) of the pulse counter card.

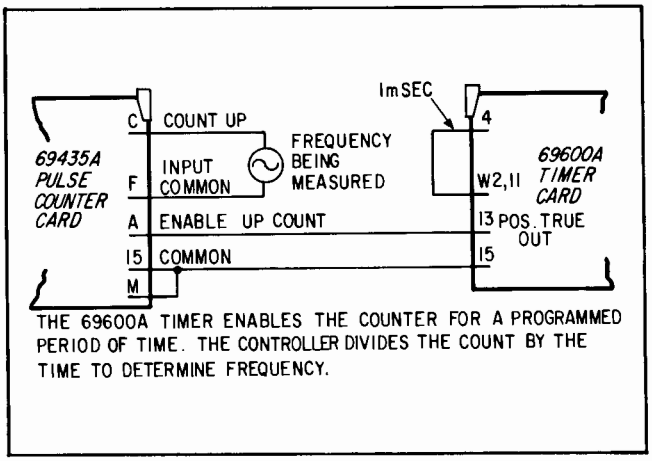


Figure 6-1. Frequency Measurement Connections

Example 63. Frequency Measurement Using 69435A and 69600A Cards

9830 Calculator Program

```
10 GOSUB 2000
20 B=10
30 C=1000
40 A=B
50 GOSUB 2400
60 CMD "?U7" ,"00040TAT"
70 FORMAT F1005.0
80 OUTPUT (13,70)"B"A2"T00160T00240TAX";
90 CMD "?5W"
100 ENTER (13,*)A
110 B=B*10
120 C=C/10
130 IF A<144 and C>1 THEN 40
140 GOSUB 2000
150 GOSUB 2200
160 A=A2*C
170 DISP "FREQUENCY ="A"HZ"
180 END
```

9820/21 Calculator Program

```
GSB "SPOLL"
10→B
1000→C
B→A
GSB "OCT"
CMD "?U7" ,"00040TAT"
FMT Z,"B",FXD*.0;WRT 13,R2
FMT Z, "T00160T00240TAX";WRT 13
CMD "?5W"
FMT *;RED 13,A
B*10→B
C/10→C
IF (A≤143) (C>1);JMP -9
GSB "SPOLL"
GSB "DEC"
R2*C→A
FXD ;PRT "FREQUENCY =",A,"HZ"
END
```

Explanation:

10 Check and clear service request
20 Variable "B" sets initial 10 millisecond gate time
30 Variable "C" sets initial scale factor for conversion to frequency in Hz as calculated in line 160
40-50 Variable "B" is converted to octal
60 Control word establishes output mode the pulse counter card registers are set to zero with the ". . . AT" statement.
70 The gate time is programmed and the counter card read as described above.
110-120 The gate time and the scale factor are adjusted to new values.
130 The total count is tested for 100_{10} (144_8) and the gate time is tested for 1 second or less. If the count is less than 100 and the gate time is less than 1 second, the program returns to line 40 and a new measurement is made.
140 Clear the service request
150-170 The total count is converted from octal to decimal, scaled appropriately, and displayed (printed).

6-284 Time Interval Measurement Using the 69435A and 69601A Cards

6-285 The following program illustrates using a 69435A pulse counter card and a 69601A frequency reference card to measure the period of a single positive pulse, and then displaying the period (or time interval) on the 9830A calculator or printing the period on the 9820A/21A calculator.

6-286 Example 64 assumes that the pulse counter card is installed in slot 401 (A) and the frequency reference card is installed in slot 402 (B). For the connections and calculations used in this particular example, the pulse duration should not exceed 40 milliseconds.

6-287 A primary consideration when measuring time intervals with the technique described here is the capacity of the pulse counter card. The user must select a reference frequency so that the maximum count capacity of 4095 is not exceeded during the longest time interval measurement capability varies inversely with the reference frequency. For example, selection of a 1kHz reference frequency would permit measurement of time intervals with durations to approximately 4 seconds. Lower reference frequencies reduce the resolution of the measurement, however, so that some compromise is necessary either way. Longer intervals may of course be measured at high resolution by cascading counter cards in order to eliminate the overflow problem.

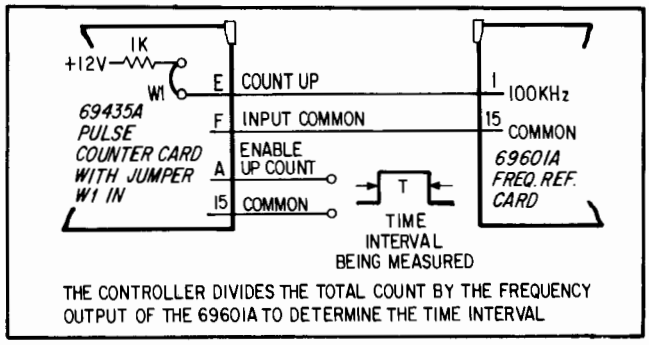


Figure 6-2. Time Interval Measurement Connections

6-288 Figure 6-2 shows the connections appropriate for the particular example described. The frequency reference card edge connector pins 1 and 15 should be connected to pulse counter card pins E and F respectively. Pulse counter card pins M and 15 should be jumpered together and the pulse to be measured should be applied between pins A and 15 of the pulse counter card. When the program in example 64 is run it will stop with "ENTER PULSE" displayed on the calculator. The user may now apply a positive pulse up to 40 milliseconds long to the pulse counter card. Depressing CONT and EXECUTE on a 9830 calculator or RUN PROGRAM on 9820/21 calculators will cause the calculator to display (or print) the time interval of the pulse.

Example 64. Time Interval Measurement Using 69435A and 69601A Cards:

9830A Calculator

```
10 GOSUB 2000
20 CMD "?U7" ,"00040TAT"
30 DISP "ENTER PULSE"
40 STOP
50 CMD "?U7" ,"00240TAX" ,"?5W"
60 ENTER (13,*)A
70 GOSUB 2200
80 A=A2/100000
90 FIXED 6
100 DISP "PULSE TIME ="A"SECONDS"
110 END
```

9820/21A Calculator Program

```
GSB "SPOLL"
CMD "?U7" ,"00040TAT"
DSP "ENTER PULSE"
STOP
CMD "?U7" ,"00240TAX" ,"?5W"
FMT *;RED 13,A
GSB "DEC"
R2/100000→A
FXD 6
PRT "PULSE TIME =",A,"SECONDS"
END
```

Explanation:

10 Check and clear service request

20 Control word establishes output mode. The pulse counter card registers are set to zero with the ". . . AT" statement.

50 Control word establishes input mode. The pulse counter card is read without a gate ("X") and the multiprogrammer system is addressed as a talker.

60-100 The data from the pulse counter card is read into the calculator, converted to decimal, scaled appropriately and displayed (printed).

Appendix A
NUMBER THEORY AND UTILITY SUBROUTINES

A-1 This appendix is divided into two major sections: (1) a brief discussion of number theory as it relates to use of the multiprogrammer system; and (2) a listing of utility subroutines which may be called from a main program to perform frequently required conversions or operations.

A-2 NUMBER THEORY

A-3 The decimal, the octal, and the binary number systems are all used to define, in numerical fashion, the data transferred within a calculator based multiprogrammer system. In addition, the user may desire to interface the multiprogrammer to external devices which send or receive data in binary coded decimal (BCD) form. Subroutines listed in Paragraph A-50 of this appendix will perform the various conversions required as part of a typical software system, but do not provide a substitute for a thorough understanding of the data code environment within which the system operates.

A-4 Decimal Numbers

A-5 Most of us are so familiar with the decimal number system that we have become largely unconscious of the underlying principles; at least so far as they relate the decimal system to other systems such as binary or octal. Any number system has what is called a "base", which is the number of unique symbols used in the particular system. Since it has ten unique symbols (0, 1, 2, - - - 9), the decimal system is the base 10 number system. Although it is often omitted since the number system is usually implied by the context, the base may be specified by a subscript:

$$235_{10} \qquad (1)$$

A-6 This subscript simply tells us that the preceding number is a unique quantitative value presented in decimal form. If we examine the individual digits that make up a decimal number, it becomes clear that the number can be represented as follows:

$$\begin{array}{rl} 235_{10} = & 2 \times 10^2 = 2 \times 100 = 200 \\ & 3 \times 10^1 = 3 \times 10 = 30 \\ & 5 \times 10^0 = 5 \times 1 = \underline{\ \ 5} \\ & \qquad\qquad\qquad 235_{10} \end{array} \qquad (2)$$

A-7 From (2) it may be seen that each digit is weighted by a power of the base (in this case 10), and that the power increases by one for each successive digit to the left. This relationship holds true for number systems to any base, and provides a longhand method for converting to base 10 from any other base.

A-8 Binary Numbers

A-9 The binary system, which is the system of machine language, is the base 2 number system, and has two unique symbols; "0" and "1".

$$11101011_2 \qquad (3)$$

A-10 If we change (3) to the form used in (2), we get:

$$\begin{array}{rl} 11101011_2 = & 1 \times 2^7 = 1 \times 128 = 128 \\ & 1 \times 2^6 = 1 \times 64 = 64 \\ & 1 \times 2^5 = 1 \times 32 = 32 \\ & 0 \times 2^4 = 1 \times 16 = 0 \\ & 1 \times 2^3 = 1 \times 8 = 8 \\ & 0 \times 2^2 = 0 \times 4 = 0 \\ & 1 \times 2^1 = 1 \times 2 = 2 \\ & 1 \times 2^0 = 1 \times 1 = \underline{\ \ 1} \\ & \qquad\qquad\qquad 235_{10} \end{array} \qquad (4)$$

A-11 Octal Numbers

A-12 The octal number system has eight unique symbols (0 through 7) and functions in a similar fashion:

$$353_8 \qquad (5)$$

and:

$$\begin{array}{rl} 353_8 = & 3 \times 8^2 = 3 \times 64 = 192 \\ & 5 \times 8^1 = 5 \times 8 = 40 \\ & 3 \times 8^0 = 3 \times 1 = \underline{\ \ 3} \\ & \qquad\qquad\quad 235_{10} \end{array} \qquad (6)$$

A-13 The octal number system has another characteristic which makes it particularly useful to the programmer. Returning to binary numbers for a moment, we see that

three binary digits allow us to represent eight unique values:

$$\begin{aligned}
111 &= (1 \times 2^2) + (1 \times 2^1) + (1 \times 2^0) = 7 \\
110 &= (1 \times 2^2) + (1 \times 2^1) + 0 = 6 \\
101 &= (1 \times 2^2) + 0 + (1 \times 2^0) = 5 \\
100 &= (1 \times 2^2) + 0 + 0 = 4 \\
011 &= 0 + (1 \times 2^1) + (1 \times 2^0) = 3 \\
010 &= 0 + (1 \times 2^1) + 0 = 2 \\
001 &= 0 + 0 + (1 \times 2^0) = 1 \\
000 &= 0 + 0 + 0 = 0
\end{aligned} \qquad (7)$$

A-14 Since a single octal digit may also represent eight unique values, we can use octal digits to represent binary triads for any binary number regardless of its size. Conversion is by direct inspection:

$$\underbrace{101}_{5}\ \underbrace{111}_{7}\ \underbrace{100}_{4}\ \underbrace{011}_{3}{}_{2} = {}_{8} \qquad (8)$$

The reverse process is just as simple:

$$\begin{matrix} 7 & 6 & 1 & 5 \\ | & | & | & | \\ 111 & 110 & 001 & 101 \end{matrix}{}_{2} = {}_{8} \qquad (9)$$

A-15 The octal number is clearly easier to remember than its binary equivalent, yet is also readily converted to or derived from binary and is therefore more closely related to machine language than decimal.

A-16 Conversion from binary to octal by inspection is a particularly useful technique when TTL output or relay cards are used as switches in a calculator based multiprogrammer system. The octal equivalent of the binary value corresponding to a desired switch pattern will produce that pattern when combined with the card slot address and output directly to the multiprogrammer via the HP-IB and the 59500A.

A-17 Decimal Conversion Algorithms

A-18 Integer conversions between the decimal and either the binary or the octal systems are less obvious and considerably more tedious. Direct conversions from decimal to binary, and vice-versa, are not usually performed, but instead conversions between decimal and octal are used in conjunction with the conversion by inspection techniques shown in (8) and (9) to achieve the same results indirectly.

A-19 Octal To Decimal. Conversion from octal to decimal may be performed as follows:

1. Multiply the most significant octal digit by 8.

2. Add the next most significant octal digit and multiply the sum by 8.

3. Repeat step 2 until the least significant digit is reached.

4. Add the least significant octal digit but do not multiply the sum by 8.

For example:

$$\begin{aligned}
&5743_8 \\
\times\ &8 \\
&40 \\
+\ &7 \\
&47 \\
\times\ &8 \\
&376 \\
+\ &4 \\
&380 \\
\times\ &8 \\
&3040 + 3 = 3043_{10}
\end{aligned} \qquad (10)$$

A-20 If we put (10) in a more general form, its relationship to (6) becomes apparent:

$$\begin{aligned}
&ABCD_Y = \\
&\times Y \\
&AY \\
&+ B \\
&AY + B \\
&\times Y \\
&AY^2 + BY \\
&+ C \\
&AY^2 + BY + C \\
&\times Y \\
&AY^3 + BY^2 + CY + D_{10}
\end{aligned} \qquad \begin{aligned}
&(A \times Y^3) + (B \times Y^2) + \\
&(C \times Y^1) + (D \times Y^0)_{10} = \\
&AY^3 + BY^2 + CY + D_{10}
\end{aligned} \qquad (11)$$

A-21 Decimal To Octal. Conversion from decimal to octal may also be reduced to a simple algorithm:

1. Divide the decimal number by 8, and write down the remainder.

2. Divide the quotient of step 1 by 8 and write down the remainder.

3. Repeat step 2 until the quotient is zero (the last remainder is also retained).

A typical example:

$$
\begin{array}{rll}
8\,)3981 & & \\
8\,)497 & \text{r.} & 5 \\
8\,)62 & \text{r.} & 1 \\
8\,)7 & \text{r.} & 6 \\
0 & \text{r.} & 7
\end{array}
\qquad (12)
$$

$$7\ 6\ 1\ 5_8$$

A-22 Several operations are implied in (12) and if shown would appear as follows:

$$
\begin{array}{rll}
8\,)3981_{10} & & \\
8\,)497 & \longrightarrow 5 \times 10^0 = & 5 \\
8\,)62 & \longrightarrow 1 \times 10^1 = & 10 \\
8\,)7 & \longrightarrow 6 \times 10^2 = & 600 \\
0 & \longrightarrow 7 \times 10^3 = & 7000 \\
 & & 7615_8
\end{array}
\qquad (13)
$$

A-23 When used in the form given in (13) this algorithm is fully reversible:

$$
\begin{array}{rll}
10\,)7615_8 & & \\
10\,)761 & \longrightarrow 5 \times 8^0 = & 5 \\
10\,)76 & \longrightarrow 1 \times 8^1 = & 8 \\
10\,)7 & \longrightarrow 6 \times 8^2 = & 384 \\
0 & \longrightarrow 7 \times 8^3 = & 3584 \\
 & & 3981_{10}
\end{array}
\qquad (14)
$$

A-24 Conversion Subroutines

A-25 The decimal to octal and the octal to decimal subroutines given in Paragraph A-50 use this same conversion technique. For reasons that will be explained shortly, the decimal to octal subroutine has been modified by the addition of program lines 2410 through 2430 for the 9830A calculator and corresponding lines of subroutine "OCT" for the 9820/21. These statements permit the subroutines to accept non-integer decimal values in the range of $-2048 \leqslant n \leqslant 10^{89}$. Otherwise the programs are identical except that the values of the constant operators are interchanged.

A-26 It is perhaps worth noting that this approach is general purpose for number systems to a positive integer base; the only constraint is that conversion must be either to or from decimal. Conversion from binary to octal, for instance, must be accomplished in two steps: (a) convert from binary to decimal and (b) convert from decimal to octal.

A-27 Negative Numbers

A-28 A number system is not really complete unless it makes some provision for negative values. The decimal system does so through the use of two additional symbols: the plus (+) and the minus (–) signs. Obviously, these two symbols serve a dual purpose since they not only indicate polarity, but also act as arithmetic operators. Any value within the range of $-\infty < n < +\infty$ may be more or less conveniently represented by this system of twelve symbols (i. e., the digits 0-9 and the "+" and "–" signs).

A-29 The relationship between the polarity indicating and the operative functions of the "+" and "–" signs is implicit in subtraction:

$$
\begin{array}{rl}
 & 10 \\
- & \underline{5} \\
 & 5
\end{array}
\quad \text{or} \quad 10 - 5 = 5 \qquad (15)
$$

is the same as:

$$10 + (-5) = 5 \qquad (16)$$

The sum of any number and its true negative is, of course, zero.

A-30 **Two's Complement Numbers.** The binary number system is useful precisely because it has only two symbols, and as a consequence, polarity cannot be indicated as it is in the decimal system without destroying the very property that permits implementation of binary codes in hardware. This reality together with the fact that from a hardware standpoint, it is generally easier to add than to subtract directly has encouraged use of the two's complement system for representing negative numbers in binary. The two's complement of any binary number is formed by:

1. Complementing each digit (changing all "0's" to "1's" and vice-versa).
2. Adding one.

Thus:

$$
\begin{array}{rl}
 & 011010011101 \longleftarrow \text{Original Number} \\
 & \downarrow \\
 & 100101100010 \longleftarrow \text{One's Complement} \\
+ & \underline{\qquad\qquad 1} \\
 & 100101100011 \longleftarrow \text{Two's Complement}
\end{array}
\qquad (17)
$$

A-31 Subtraction using two's complement numbers is as follows:

$$
\begin{array}{rl}
12_{10} & 1100 \quad \text{(Two's complement} \\
-\ \underline{9_{10}} & +\ \underline{0111} \longleftarrow \text{of } 1001_2) \\
3_{10} & 10011
\end{array}
\qquad (18)
$$

(Last carry is ignored) $\qquad 0011_2 = 3_{10}$

A-32 Returning to (17), we see that adding the original number and its two's complement produces:

$$\begin{array}{r} 011010011101 \\ +\ 100101100011 \\ \hline 1\ 000000000000 \end{array} \qquad (19)$$

$$000000000000_2 = \text{zero}$$

A-33 The result is zero and demonstrates the validity of treating the two's complement as a true negative of its root binary number.

A-34 At this writing, there are only two multiprogrammer cards which require manipulation of negative numbers: the 69321B Voltage D/A Card and the 69421A Voltage Monitor A/D Card.

A-35 69321B D/A Card

A-36 The 69321B card is a twelve bit bipolar (two's complement) digital to analog converter and may be programmed to output a voltage in the range of −10.240 through +10.235 volts. The twelve bit binary word accepted by the D/A module allows for 4096_{10} ($2^{12} = 4096$) unique programmed codes. Since zero is one of the possibilities, however, the actual decimal values fall in the range of 0 through 4095. Presented graphically, the range of output voltages and the corresponding decimal, octal, and binary values appear as shown in Figure A-1.

A-37 If we take the absolute value of negative full scale and add it to positive full scale (10.240 + 10.235), we arrive at a value of 20.475 volts as the actual range of the module. This figure may then be divided by 4095_{10} to yield the analog equivalent value of the least significant bit (LSB), which in this case is exactly 0.005 volts or 5mV. All nominal output voltages will be an integer multiple of this 5mV value. Note that we divide by 4095 rather than 4096 since a range of X^n (2^{12}) values has X^n-1 ($2^{12}-1$ or 4095_{10}) intervals.

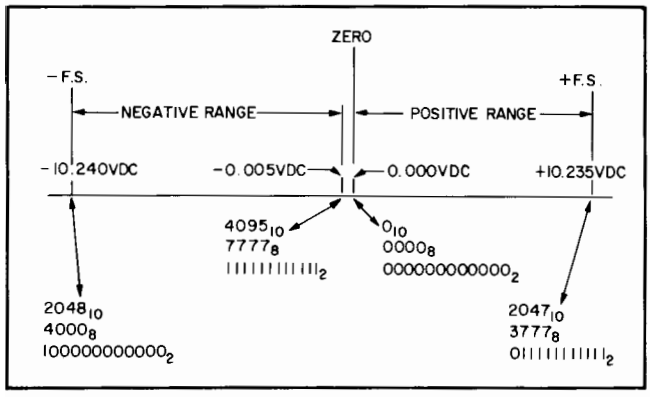


Figure A-1. D/A Voltage Converter Card, Output Range

A-38 Armed with the knowledge that the LSB is 5mV, we may now divide any voltage value within the range of the converter by that figure and obtain the decimal equivalent of the binary code that will produce the desired output. For example:

$$\frac{5.000V}{.005V} = 1000_{10} \qquad (20)$$

$$-\frac{6.745V}{.005V} = -1349_{10} \qquad (21)$$

$$\frac{0.123V}{.005V} = 24.6 \qquad (22)$$

A-39 The decimal equivalent values derived by the method shown in (20), (21), and (22) will always fall within the range of -2048_{10} to $+2047_{10}$ provided that the voltage to be programmed is within the specified range of −10.240 to +10.235 volts. The additional program statements included in the decimal to octal subroutine permit the program to accept negative and/or non-integer values by automatically rounding-off and scaling inputs to integer values within the range of 0_{10} to 4095_{10}. The round-off routine ignores the sign of the number (+1.5 becomes + 2 and −1.5 becomes −2) and is accurate to nine decimal places. The scaling routine converts negative values to the positive decimal equivalent of the two's complement code corresponding to the desired negative voltage.

A-40 69421A A/D Card

A-41 The conversions required by the 69421A voltage monitor card are basically the inverse of those just discussed. The process is simplified, however, since the octal value returned to the calculator by the 69421A card is always an integer in the range of 0_8 through 7777_8. The octal to decimal subroutines given in Paragraph A-50 convert this value to a decimal number in the range of 0 through 4095. Since the 69421A is a bipolar device and produces a two's complement code, the decimal number must be scaled and then multiplied by the analog equivalent value of the LSB in order to yield the actual decimal value of the applied voltage. This conversion is accomplished as follows:

1. If the decimal value is 2048_{10} or larger then subtract 4096_{10}; if not, go directly to step 2.

2. Multiply the decimal value from step 1 by the analog equivalent value of the LSB.

For example:

$$6065_8$$
$$\downarrow$$
$$3125_{10}$$
$$-\ 4096_{10}$$
$$-\ 971_{10}$$
$$\times\ .005 \text{ volts}$$
$$-\ 4.855 \text{ volts} \qquad (23)$$

A-42 The sample programs for the 69421A given in Chapter VI show the appropriate statements for the operations performed in (23) (e. g., line numbers 60 through 80 for the 9830A calculator).

A-43 BCD Conversions

A-44 Also included in Paragraph A-50 are subroutines for converting from decimal to BCD - weighted octal and the reverse. These programs are primarily intended to permit transfer of numerical data to and from BCD programable external devices via the 69331A digital output and the 69431A digital input cards. The algorithm for these conversions is similar to that used for the base conversion routines discussed previously.

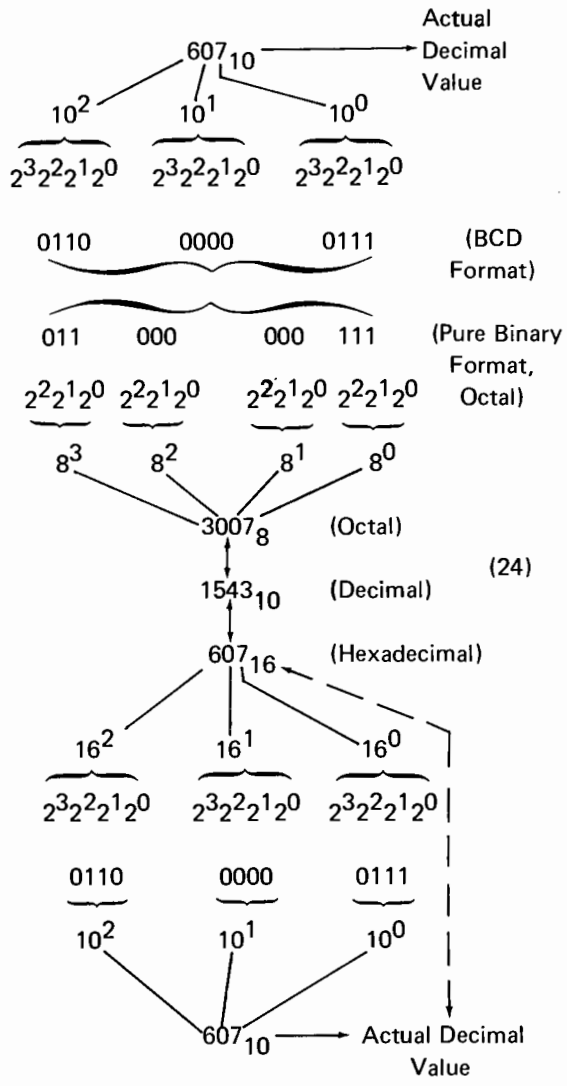
A-45 BCD To Decimal. For BCD to decimal conversions, the octal value corresponding to the twelve bit pattern produced by three decades of BCD data is first converted to decimal. This decimal value is BCD - weighted, however, and must undergo further conversion before the encoded actual decimal value is recovered. Since decimal equivalents of BCD codes and the hexadecimal (base 16) system are both higher order representations of binary quads (BCD is a truncated version of hexadecimal in the sense that it makes use of just ten of the sixteen unique combinations in a binary quad), the weighted decimal value will yield the actual decimal value when converted to base sixteen. The latter portion of the BCD to decimal subroutine performs this operation.

NOTE

The technique outlined here is based upon the assumption that an octal value representing a valid BCD code is input to the subroutine. Octal representations of full hexadecimal or invalid BCD codes will produce erroneous results.

A-46 Decimal To BCD. The decimal to BCD conversion is the inverse of the subroutine just described. Decimal numbers in the range of 0 through 999 may be converted to a BCD - weighted octal code using this program. The octal value will produce a twelve bit pattern corresponding to three decades of BCD coded numerical data.

A-47 The relationships between the various number systems are summarized for a decimal value of 607 in (24).



Note that the change from BCD to pure binary format is a conversion which takes place in the hardware. It is this change in format that permits us to interpret what is actually a hexadecimal equivalent value (607_{16}) as a decimal value; we can in effect justify an implied conversion as a compensation for the hardware.

A-48 The algorithm as shown in (24) also takes advantage of the noncontiguous nature of BCD coding. Since we have limited the permissible inputs to valid BCD codes, certain octal, and likewise certain decimal, values do not occur. Conveniently, these particular values are the only ones within the twelve bit range of the multiprogrammer which would produce one or more of the hexadecimal characters A through F. The portion of the subroutine which converts from decimal to hexadecimal (or vice-versa) will not handle

these values correctly, however, the constraints outlined above eliminate the need to do so. No provision has been made to detect and flag invalid BCD codes, but this capability may easily be added by the user if desired. The program statements shown below provide invalid code detection in the 9830A subroutine; the 9820/21A program may be modified in a similar fashion.

```
3060 IFA2 – (INT(A2/256)*256) < 154 THEN 3062
3061 GOTO 4000
3062 IF A2 – (INT(A2/16) * 16) < 10 THEN 3064
3063 GOTO 4000
3064 IF A2 > 2457 THEN 4000
3065 A = A2
4000 REM ERROR SUBROUTINE
```

A-49 **Large BCD Numbers.** Larger blocks of BCD data may be transferred in three decade slices via multiple cards if appropriate scaling operations are performed. The outline below illustrates the technique for an input transfer of nine decades (N is the recovered actual decimal value).

1. Read data from card #1 and store in A.
2. Go to BCD to decimal subroutine.
3. N = A_2.
4. Read data from card #2 and store in A.
5. Go to BCD to decimal subroutine.
6. N = (10 ↑ 3 * A_2) + N.
7. Read data from card #3 and store in A.
8. Go to BCD to decimal subroutine.
9. N = (10 ↑ 6 * A_2) + N.

A-50 UTILITY SUBROUTINES

A-51 Table A-1 lists the utility subroutines, labels their function, and indicates the entry and exit variables used to connect the subroutine to the main program. The actual program listings for both the 9830A and the 9820/21A calculators follow in order.

Table A-1. Subroutine Summary

Subroutine		Function	Entry Variable		Exit Variable	
9830	**9820/21**		**9830**	**9820/21**	**9830**	**9820/21**
2000	"SPOLL"	Serial Poll	None	None	A4	R4
2200	"DEC"	Octal To Decimal Conversion	A	A	A2	R2
2400	"OCT"	Decimal To Octal Conversion (–2048 TO + 4095)	A	A	A2	R2
2600	"BIT"	Print Bits for Card Test Programs	A2	R2	None	None
2800	"D-BCD"	Decimal To BCD Weighted Octal Conversion	A	A	A2	R2
3000	"BCD-D"	BCD Weighted Octal To Decimal Conversion	A	A	A2	R2

9830A Subroutines

```
2000 REM SERIAL POLL ROUTINE
2010 CMD "?U"
2020 FORMAT 5B
2030 OUTPUT (13,2020)256,95,53,24,512;
2040 CMD "W"
2050 A4=RBYTE13
2060 CMD "?U"
2070 FORMAT 3B
2080 OUTPUT (13,2070)256,25,512;
2090 RETURN
```

```
2200 REM OCTAL TO DECIMAL CONVERSION
2210 A1=A2=0
2220 A2=A2+(A-INT(A/10)*10)*8↑A1
2230 A=INT(A/10)
2240 A1=A1+1
2250 IF A#0 THEN 2220
2260 RETURN
```

9830A Subroutines (Cont.)

```
2400 REM DECIMAL TO OCTAL CONVERSION
2410 IF A>-0.5 THEN 2430
2420 A=4096+A-1E-08
2430 A=INT(A+0.5)
2440 A1=A2=0
2450 A2=A2+(A-INT(A/8)*8)*10↑A1
2460 A=INT(A/8)
2470 A1=A1+1
2480 IF A#0 THEN 2450
2490 RETURN



2600 REM DECIMAL TO BIT CONVERSION
2610 IF A2#0 THEN 2640
2620 DISP "NO DATA RECEIVED"
2630 GOTO 2710
2640 FOR A1=11 TO 0 STEP -1
2650 A=2↑A1
2660 IF A2-A>-1 THEN 2680
2670 GOTO 2700
2680 DISP "BIT"A1"PIN"A1+1
2690 A2=A2-A
2700 NEXT A1
2710 RETURN



2800 REM DECIMAL TO BCD CONVERSION
2810 A1=A2=0
2820 A2=A2+(A-INT(A/10)*10)*16↑A1
2830 A=INT(A/10)
2840 A1=A1+1
2850 IF A#0 THEN 2820
2860 A=A2
2870 A1=A2=0
2880 A2=A2+(A-INT(A/8)*8)*10↑A1
2890 A=INT(A/8)
2900 A1=A1+1
2910 IF A#0 THEN 2880
2920 RETURN



3000 REM BCD TO DECIMAL CONVERSION
3010 A1=A2=0
3020 A2=A2+(A-INT(A/10)*10)*8↑A1
3030 A=INT(A/10)
3040 A1=A1+1
3050 IF A#0 THEN 3020
3060 A=A2
3070 A1=A2=0
3080 A2=A2+(A-INT(A/16)*16)*10↑A1
3090 A=INT(A/16)
3100 A1=A1+1
3110 IF A#0 THEN 3080
3120 RETURN
```

9820/21A Subroutines

```
0:
"SPOLL"; CMD "?→S
S"⊢
1:
CMD "W"; RDB 13→R
4⊢
2:
CMD "?→X"⊢
3:
RET ⊢
R400



4:
"DEC"; 0→R1→R2⊢
5:
R2+(A-INT (A/10)
*10)*8↑R1→R2⊢
6:
INT (A/10)→A⊢
7:
R1+1→R1⊢
8:
IF A≠0; JMP -3⊢
9:
INT (R2+.005)→R2
⊢
10:
RET ⊢
R388



11:
"OCT"; 0→R1→R2⊢
12:
IF A>-.5; JMP 2⊢
13:
4096+A-1E-8→A⊢
14:
INT (A+.5)→A⊢
15:
R2+(A-INT (A/8)*
8)*10↑R1→R2⊢
16:
INT (A/8)→A⊢
17:
R1+1→R1⊢
18:
IF A≠0; JMP -3⊢
19:
RET ⊢
R373
```

```
25:
"BIT";FXD ;IF R2
≠0;JMP 3⊢
21:
PRT "NO DATA REC
EIVED"⊢
22:
JMP 8⊢
23:
11→R1⊢
24:
2↑R1→A⊢
25:
IF R2-A>-.5;JMP
2⊢
26:
JMP 3⊢
27:
PRT "BIT",R1,"PI
N";R1+1⊢
28:
R2-A→R2⊢
29:
R1-1→R1;IF R1>-1
;JMP -5⊢
30:
RET ⊢
R355
```

```
31:
"D-BCD";0→R1→R2⊢
32:
R2+(A-INT (A/10)
*10)*16↑R1→R2⊢
33:
INT (A/10)→A⊢
34:
R1+1→R1⊢
35:
IF A≠0;JMP -3⊢
36:
INT (R2+.005)→A⊢
37:
0→R1→R2⊢
```

```
38:
R2+(A-INT (A/8)*
8)*10↑R1→R2⊢
39:
INT (A/8)→A⊢
40:
R1+1→R1⊢
41:
IF A≠0;JMP -3⊢
42:
INT (R2+.005)→R2
⊢
43:
RET ⊢
R332
```

```
44:
"BCD-D";0→R1→R2⊢
45:
R2+(A-INT (A/10)
*10)*8↑R1→R2⊢
46:
INT (A/10)→A⊢
47:
R1+1→R1⊢
48:
IF A≠0;JMP -3⊢
49:
INT (R2+.005)→A⊢
50:
0→R1→R2⊢
51:
R2+(A-INT (A/16)
*16)*10↑R1→R2⊢
52:
INT (A/16)→A⊢
53:
R1+1→R1⊢
54:
IF A≠0;JMP -3⊢
55:
INT (R2+.005)→R2
⊢
56:
RET ⊢
R308
```

HEWLETT hp PACKARD

92500-90003

Printed in U.S.A.