PICTURE SYSTEM 2 HARDWARE
DIAGNOSTIC MANUAL

Evans & Sutherland Computer Corporation

November 1978

Customer Engineering Dept.
Evans & Sutherland Computer Corporation
580 Arapeen Drive
Salt Lake City, Utah 84108

First Edition May 1977

Second Edition November 1978

All reference to this document should be made to:

No. 901180-001

TABLE OF CONTENTS

CHAPTER 5 PICTURE SYSTEM II DIAGNOSTICS AND MAINTENANCE

RUN FROM CONSOLE ONLY

CHAPTER 6 QSDDT OPERATION

LIST OF APPENDICES

PREFACE

0.1 MANUAL OBJECTIVES

This manual contains the general information required to utilize diagnostic programs for the PICTURE SYSTEM 2 (PS2) manufactured by Evans & Sutherland Computer Corporation, when interfaced to the Digital Equipment Corporation (DEC) PDP-11 Computer.

0.2 OPERATING SYSTEMS

Picture System diagnostics are distributed for execution on the PDP-11 under the following operating systems: (1) E&S Diagnostic Monitor (ESD). This package includes diagnostics for PICTURE SYSTEM 1 as well as PICTURE SYSTEM 2. Executable programs and all other files in this package are RT-11 compatible. (2) Unmapped RSX-11M, which does not utilize PDP-11 memory management (3) Mapped RSX-11M, which does utilize PDP-11 memory management. At the present time, only PS-2 diagnostics which do not utilize DMA's or interrupts are available in this package.

0.3 STRUCTURE OF THE DOCUMENT

Chapter one details startup procedures for the three packages mentioned above.

Chapter two describes the E&S Diagnostic Monitor, its operating environment, the commands used for operator communications and the errors which the operator may encounter when using the E&S Diagnostic Monitor.

Chapter three describes the E&S Diagnostic File Utility Program (UPDATE) operation.

Chapter four describes the Patch Utility Program (PATCH) operation.

Chapter five describes the PS-2 Diagnostic Programs (QSD000-QSD116) and Acceptance Test Programs (RSD000-RSD012). It includes an overview of this collection of programs, general operating procedures, and a functional description of each program. The information in this chapter is, for the most part, applicable to operation on the PDP-11 as well as all available processors other than the PDP-11.

Chapter six describes the operation of QSDDT, a general purpose troubleshooting program for PICTURE SYSTEM 2. As in chapter five, the information in this chapter is for the most part computer independent.

Appendix A contains information concerning available distribution media and operating systems for the PICTURE SYSTEM 2 diagnostics on the PDP-11.

Appendices B and C summarize the information in chapters two through four.

Appendices D and E provide, in conjunction with sections 1.4 and 1.5, RSX-11M operating information.

Appendices F through I contain miscellaneous reference information required for operation of the diagnostics.

0.4 RELATED DOCUMENTS

The PICTURE SYSTEM 2/PDP-11 Reference Manual (E&S) contains a programmer's functional descriptions of the PS-2 hardware, and other general programming information.

The PICTURE SYSTEM 2 Maintenance Manual (E&S) contains detailed hardware information and theory of operation for PICTURE SYSTEM 2.

The PICTURE SYSTEM 2 User's Manual (E&S) contains an introduction to computer graphics, and a user's description of the PICTURE SYSTEM 2 Graphics Software Package.

The PICTURE SYSTEM 2 Acceptance Tests (E&S) describes the PS-2 Acceptance Test procedures, and related use of the Acceptance Test Programs RSD000 throgh RSD012.

RSX-11M OPERATION: various documents published by Digital Equipment Corporation contain helpful information, as outlined in Appendix D of this manual.

CHAPTER ONE

1.0 STARTUP PROCEDURE

For the purposes of this chapter, the distributions of the E&S Diagnostic Package, as outlined in Appendix A, need be distinguished only by distribution medium: RK05 Disk, DECtape, paper tape, or Magtape (TU10 or TU16). For all but the last of these cases, the distribution medium (corresponding to the load device) is also the system medium, and resides upon the system device. Thus, in the first three cases, the startup procedure consists merely of bootstrapping the system device, and Section 1.1 may be ignored.

For distributions on Magtape, however, the system device will be some type of disk drive, and it will be necessary to transfer the system image from Magtape to the system device, as explained in Section 1.1. Thereafter, the system device may be booted as explained in subsequent sections.

1.1 TRANSFERRING THE SYSTEM IMAGE FROM MAGTAPE TO SYSTEM DEVICE

Mount a scratch disk pack on drive zero. Any data on this disk will be destroyed by the procedure which follows. Mount the Magtape containing the E&S Diagnostic System on the Magtape drive zero. The disk image has a label (e.g. ESPS2DSRSX) which should be noted.

Bootstrap the Magtape, using a ROM Bootstrap if available (see instructions in the appropriate subsection of 1.3), or the keyed-in program for TU10 Magtapes in Section 1.3.3. The following message should appear on the system console:

```
RSX-11S V02 BL18
VOLUME PRESERVATION PROGRAM - V02.4
FOR HELP TYPE /HE
nn. BLOCKS AVAILABLE FOR BUFFERING

PRE>
```

In the following instructions, XX: represents a mnemonic for the system device (disk) as follows:

DK: RK05
DB: RP04, RP05, RP06
DP: RP02, RP03
DX: RX01 Floppy Disk

Also, YY: represents the mnemonic for the Magtape unit as follows:

MM: TU16
MT: TU10

If necessary, format the scratch disk, using a command of the following form:

```
PRE>XX0:/FO
```

Transfer the disk image from Magtape onto the scratch disk pack using a command of the following form:

```
PRE>XX0:/VE=YY0: "label"
```

Where "label" is the disk image label mentioned above.

PRESRV will type a message asking the operator to mount the input volume then type <CR>. When the transfer has finished, PRESRV will type:

```
"*** BEGIN VERIFICATION"
"MOUNT INPUT VOLUME AND TYPE CR"
```

Enter a carriage-return and PRESRV will read and compare the contents of Magtape and disk pack. An error message

indicating insufficient buffer space for verification may occur, in which case verification may be omitted.

Assuming no errors in the data transfer, the system device may now be bootstrapped as explained below. The distribution Magtape should be set aside as a backup.

1.2 BOOTSTRAPPING THE SYSTEM DEVICE

The operating system (or Paper Tape Loader)for the E&S Diagnostic Packaged is loaded into computer memory and executed by "bootstrapping" the system device, which may be any of the following:

DB: RP04, RP05, RP06 Disk
DK: RK05 Disk
DP: RP02, RP03 Disk
DX: RX01 Floppy Disk
DT: DECtape
PR: Paper Tape Reader

Assuming the system image has been transferred to the system device (if not, go back to Section 1.1), the following steps are required: (a) Mount the system device and put it on-line, and (b) bootstrap the device according to instructions below which apply to the available ROM Bootstrap, or by keying in and executing a bootstrap program as in Sections 1.3.1 through 1.3.3.

After the system device has been booted, diagnostic programs may be initiated and terminated by means of the supporting operating system: The E&S Diagnostic Monitor (cf Chapters 2-4) or the DEC RSX-11M V03 unmapped operating system (cf Appendix D) or the DEC Paper Tape Loader.

1.3 BOOTSTRAP PROCEDURES

If a ROM Bootstrap is available consult the pertinent instructions below to bootstrap a Magtape or System device. If no ROM Bootstrap is available, then subsections 1.3.1 - 1.3.3 give instructions for bootstrapping an RK05 Disk Pack, a DECtape, or a TU10 Magtape by means of a keyed-in program.

1.3.1 BOOTSTRAPPING THE RK05 DISK

1. Mount the system device on drive 0.
2. If the distribution includes the E&S Diagnostic Monitor be sure the WRITE PROTECT light is not lit.
3. Press HALT.
4. Set the console switches to 1000. Press LOAD ADRS.
5. Load the following program into locations 1000 through 1020 by putting each program data value into the console data switches, then lifting the DEPOSIT switch:

Address	RK05 Bootstrap Program Data
1000:	12700
1002:	177406
1004:	12710
1006:	177400
1010:	12740
1012:	5
1014:	105710
1016:	100376
1020:	5007

6. Load Address 1000 again, lift HALT, and press START.

1.3.2 BOOTSTRAPPING THE DECTAPE

1. Mount the system device on drive 0, and set the REMOTE/OFF/LOCAL switch to REMOTE.
2. If the distribution includes the E&S Diagnostic Monitor, be sure the tape drive is WRITE ENABLED.

3. Press HALT
4. Set the console switches to 1000. Press LOAD ADRS.
5. Load the following program into locations 1000 through 1040 by putting each program data value into the console data switches, then lifting DEPOSIT:

Address	DECtape Bootstrap Program Data
1000:	12700
1002:	177344
1004:	12710
1006:	177400
1010:	12740
1012:	4002
1014:	5710
1016:	100376
1020:	12710
1022:	3
1024:	105710
1026:	100376
1030:	12710
1032:	5
1034:	105710
1036:	100376
1040:	5007

6. Load Address 1000 again, lift HALT, and press START.

1.3.3. BOOTSTRAPPING THE TU10 MAGTAPE

1. Mount the Magtape containing the system image on drive 0. Be sure the following lights are lit: PWR, LOAD, SEL, RDY and LD PT.
2. Press HALT on the computer console.
3. Set the console switches to 10000. Press LOAD ADRS.
4. Load the following program into locations 10000 through 10036 by putting each program data value into the console data switches, then lifting DEPOSIT:

ADDRESS	TU10 Bootstrap Program Data
10000:	12700
10002:	172524
10004:	5310

Address	TU10 Bootstrap Program Data
10006:	12740
10010:	60011
10012:	105710
10014:	100376
10016:	5710
10020:	100767
10022:	12710
10024:	60003
10026:	105710
10030:	100376
10032:	5710
10034:	100777
10036:	5007

5. Load Address 10000 again, lift HALT, and press START.

1.3.4 BM792-YB BOOTSTRAP ROM

1. Press HALT, and Load Address 173100.
2. Enter the address of the load device in the console switches:

   RK05: 177406
   DECtape: 177344
   RP03: 176716

3. Lift HALT, press START

1.3.5 MR-11DB BOOTSTRAP ROM

1. Press HALT.
2. Enter the address of the load device in the console switches:

   RK05: 173110
   DECtape: 173120
   TU10: 173136
   RP03: 173154

3. Press LOAD ADDRESS
4. Lift HALT, press START.

1.3.6 BM873-YA BOOTSTRAP ROM

1. Press HALT.
2. Enter the address of the load device in the console switches:

   RK05: 173010
   DECtape: 173030
   TU10: 173050
   Paper Tape: 173312
   RP03: 173100

3. Press LOAD ADRS.
4. Lift HALT, press START.

1.3.7 BM873-YB BOOTSTRAP ROM

1. Press HALT.
2. Enter the address of the load device in the console switches:

   RK05: 173030
   DECtape: 173070
   TU10: 173110
   TU16: 173150
   RP04: 173320
   RP02/03: 173350
   Paper Tape: 173620

3. Press LOAD ADRS.
4. Lift HALT, press START.

1.3.8 M9301-YA OR M9301-YB BOOTSTRAP ROM

1. Press the BOOT switch.
2. In response to the "$" prompt on the console terminal, enter the load-device mnemonic:

   RP04: DB (M9301-YB only)
   RK05: DK

RP02/03:	DP
DECtape	DT
RX01 Floppy:	DX
TU16:	MM
TU10:	MT
Paper Tape:	PR

1.3.9 M9301-YC BOOTSTRAP ROM (PDP-11/70 only)

1. Press HALT, and lift HALT.
2. Set the start address 17765000 in the console switches.
3. Press LOAD ADDR.
4. Set the device unit number (should be 0) in switches 0-2.
5. Set the appropriate device code in switches 3 through 6.

TU10:	1
DECtape:	2
RK05:	3
RP02/03:	4
TU16:	6
RP04/05/06:	7

6. Switches 7 through 21 must be down (off).
7. Press START.
8. If the computer halts before typing the operating system startup message, a DEC hardware failure is indicated.

1.3.10 M9301-YF BOOTSTRAP ROM

1. Press the BOOT switch.
2. In response to the "$" prompt, enter the appropriate device code:

RP04/05/06:	DB
RK05:	DK
DECtape:	DT
RX01 Floppy:	DX
TU16:	MM
TU10:	MT
Paper Tape:	PR

1.4 TRANSFERRING DIAGNOSTIC TASKS FROM MAGTAPE TO DISK (UNMAPPED RSX-11M)

The diagnostics must be run under the Unmapped Baseline System which is part of DEC's RSX-11M operating system. The diagnostics are built to run in partition PAR14K.

Case A: Using the Customer's Multiuser Pack.

```
>SET /UIC=[XXX,XXX]
```

select a UIC under which the diagnostic tasks may be permanently stored. First do a UFD command, if necessary. Mount the "PS-2 Diagnostics Tasks" magtape.

```
>LOA MT:
```

(if necessary)

```
>FLX /RS/CO/BL:58.=MT:[220,15]*.*/DO
```

Now, with no other users on the system, bootstrap the Unmapped operating system:

```
>INS [1,54]BOO
>BOO [1,50]RSX11M
   .
   .
   .
>SET /UIC=[XXX,XXX]
```

Now the diagnostics may be run according to Appendix D of the Diagnostics Manual.

Case B: Using a scratch RPØ4 or RPØ6 Pack mount the scratch pack and the magtape labelled ES-PS2-DSØ3Ø4-MTRSX-DB. Boot the magtape, or by other means run the DEC stand-alone task, PRESRV.

```
  :
PRE>DBØ:/FO                  -  (if necessary)
  :
PRE>DBØ:/VE=MTØ:DBRSXØ3Ø4    -  (TU10 Magtape)
  or
PRE>DBØ:/VE=MMO:DBRSXØ3Ø4    -  (TU16 Magtape)
  :
```

Now bootstrap the scratch pack, by the PRESRV command "DBØ:/BO" or by using the installation bootstrap ROM.

```
  .
  .
>SET /UIC=[200,200]          (if necessary)
>INS MT:                     (if necessary)
>RUN $FLX
>FLX /RS/CO/BL:58=MT:[220,15]*.*/DO
```

Now the diagnostics may be run according to Appendix D of the Diagnostics Manual.

1.5 TRANSFERRING DIAGNOSTIC TASK FROM MAGTAPE TO DISK (MAPPED RSX-11M)

The PS-2 Diagnostics were originally designed for use on a single-user system only. A complete set of diagnostics which run under Unmapped RSX-11M is distributed under UIC=[220,15]. A subset of the PS-2 Diagnostics is available for use on a Mapped Multi-user System to avoid, when possible, shutting

down the Multi-User System.

PS-2 Diagnostics may be run on a mapped multi-user operating system provided that no DMA's or interrupts are used, and the effect of checkpointing is of no concern. The mapped diagnostic tasks do not use a driver, but rather perform Picture System I/O by direct access to the interface registers through a common block named PSDEVØ, similar to the PSDEV common block in the "fast" version of the PS2/RSX-11M Graphics Software Package (see "Getting Started With PS2/RSX-11M" section 2.3.1).

Log on as a privileged user, and set the UIC equal to a value provided by the customer for permanent storage of the mapped diagnostic tasks. This UIC should be different from the one used for unmapped diagnostics, as name conflicts will otherwise occur.

```
>FLX /RS/IM=MT:[220,14]PSDEVØ.*/DO
>SET /MAIN=PSDEVØ:7676:1:DEV
>MAC PSDEVØ=PSDEVØ
```

Where [XXX,XXX] is the selected UIC for storage of Mapped Diagnostic Tasks:

```
>SET /UIC=[1,1]
>PIP /NV=[XXX,XXX]PSDEVØ.OBJ
>TKB @[XXX,XXX]PSDEVØ
>INS PSDEVØ
>PIP PSDEVØ.OBJ;*/DE
>SET /UIC=[XXX,XXX]
>FLX /RS/CO/BL:58.=MT:[220,14]*.TSK/DO
```

Now it should be possible to run the PS-2 Diagnostics.

Note: Some phases of the available diagnostics will attempt to use DMA. If those phases are executed, undesirable results will occur. This pertains primarily to visual tests QSD020, QSD026, QSD027, and QSD028. If in doubt, consult the program description and the "HELP" messages.

For the contents of PSDEVØ.MAC and PSDEVØ.CMD, and procedures for non-standard PS-2 UNIBUS addresses, consult Appendix E, "PSDEVØ Generation and Modification".

Once the above procedure has been performed to build the common block PSDEVØ, only the "SET /MAIN" and "INS PSDEVØ" commands need to be repeated on subsequent occasions, provided PSDEVØ.TSK and PSDEVØ.STB are saved in area [1,1].

CHAPTER TWO

2.0 E&S DIAGNOSTIC MONITOR

The E&S Diagnostic Monitor is a control program designed for the DEC PDP-11 computer to provide services which are typically required for general diagnostic capability. It will execute on any PDP-11 with a minimum of 16K memory and operates as either a DECtape or Disk (DECpack) based monitor. The Diagnostic Monitor handles Disk or DECtape as file structured devices and has a file structure compatible with the DEC RT-11 operating system.

The devices supported by the E&S Diagnostic Monitor are:

DEVICE	DEVICE MNEMONIC
RK05 DISK	RKn:
DECtape	DTn:
Console Terminal	TT:
Paper Tape REader	PR:
Paper Tape Punch	PP:

The file utility program, UPDATE (see Chapter Three), provides the capability of updating diagnostic files by deletion and transfer between any of these devices.

The user of the E&S Diagnostic Monitor initiates the monitor by booting it from the system device (RK0 or DT0) as described in Chapter 1, and then interfaces with it by means of the Keyboard Commands described in Section 2.1. All errors that occur are logged on the console terminal and detailed in Section 2.2.

2.1 OPERATOR COMMUNICATION

When the E&S Diagnostic Monitor is booted, it prints the following on the console terminal:

```
E&S DIAGNOSTIC MONITOR
xxK MEMORY
*
```

Where xx = 8,12,16,24 or 28 indicating the amount of memory available on the particular hardware configuration. The * printed indicates the Diagnostic Monitor is ready to accept operator input from the console terminal keyboard.

The operator communicates to the Diagnostic Monitor by entering Keyboard commands via the console terminal. The Keyboard commands available to the operator are DATE, ASSIGN and RUN and are detailed in the following sections.

Certain keys of the console terminal are special functions under the Diagnostic Monitor. These keys and the special functions assigned to them are as follows:

CONTROL/C -- Echoes ↑C on the terminal and interrupts the execution of the diagnostic program and returns to the monitor. Two CONTROL/C's cause the currently running program to be aborted and the Diagnostic Monitor to be returned to.

CONTROL/U -- Deletes the current line and echoes ↑U at the terminal.

RUBOUT -- Deletes the last character from the current line and echoes the character deleted on the terminal.

2.1.1 DATE COMMAND

The DATE command enters the specified date to the system. This date remains in effect until a new DATE command is issued. The command is of the form:

```
*DATE     dd-mmm-yy(CR)[1]
              or
*DA       dd-mmm-yy(CR)
```

Where dd-mmm-yy is the day, month and year to be entered.

If no date is entered, then no date is assigned to file update or directory listings.

Example:

```
*DATE       1-JAN-74(CR)
               or
*DA        31-DEC-74(CR)
```

2.1.2 ASSIGN COMMAND

The ASSIGN command assigns the specified device name as alternate to the standard device name (see Section 1). The command is of the form:

```
*ASSIGN  dev adev(CR)
           or
*AS      dev adev(CR)
```

[1] The notation (CR) denotes the entry of a carriage return by the operator.

Where:

dev is a standard device name.

RKn:
DTn:
TT:
PR:
PP:

adev is a 1-3 character alternate device name.

Examples:

*ASSIGN DTØ TT (CR)

or

*AS DTØ TT (CR)

*AS(CR) (this removes all previous alternate device name assignments).

2.1.3 RUN COMMAND

The RUN command causes the specified file to be loaded into memory and starts execution at the entry point of the program. The command is of the form:

*RUN dev:file.ext(CR)

Where:

dev is one of the standard device names. If dev: is not specified, the system device is assumed.

file is a 1-6 character file name.

.ext is a 1-3 character file extention. If .ext

is not specified, the diagnostic monitor will assume an extention of the form Sxx where xx = 01-09. The device will then be searched for a file with the name file.S09, then file .S08, then file.S07,...file.S01. If none of these files are found then an error message will be printed, otherwise the file which was found will be loaded and executed. This searching sequence is in keeping with the E&S program naming convention where the last two digits of the extension indicate the revision level of the program. See Section 3.1 for further details.

Examples:

*RUN DTØ:PSP002.S02

or

*RU PSP002.S02

or

*RU PSD100 (If the system device contains the programs PSD100.S01 and PSD100.S02, the latter PSD100.S02 would be run by this command.)

2.2 ERRORS

The following error messages may be output by the E & S Diagnostic Monitor.

ERROR MESSAGE	MEANING
HUH?	Invalid command or command string parameters.
NO FILE	File name not found or incorrectly specified in command string.

NO ROOM	Attempt to RUN a file which is too big for the available memory.
NO WAY	Unable to close the file without the handler available.
NO HANDLER	Device handler unavailable for I/O request.
DEVICE FULL	No room in device directory for further entries.
SYS ERROR	I/O error on system device. Retry the operation.
I/O ERR n	Fatal I/O error (n denotes type of operation). Restart the operation by RUNning the program again or rebooting the Diagnostic Monitor.

CHAPTER THREE

3.0 FILE UTILITY PROGRAM (UPDATE)

UPDATE is the file utility program for the E&S Diagnostic Monitor. It provides the facilities to initialize device directories, delete and transfer files and other utility functions useful in updating and maintaining E&S diagnostics. The following sections detail the facilities of UPDATE.

3.1 USING UPDATE

To use UPDATE, one must request the Diagnostic Monitor to run it from the system (or other specified) device:

```
*RUN UPDATE(CR)
```

When UPDATE begins execution, it prints the following on the console terminal:

```
UPDATE.Sxx      (xx denotes the current version number)
E&S DIAGNOSTIC UTILITY
>
```

The > indicates that UPDATE is ready to receive a user command.

A user command consists of:

1. an operation
2. a file name and an operation
3. several file names and an operation

FILENAME CONVENTIONS

All files maintained under the E&S Diagnostic Monitor must be of the form:

filename.ext

Where:

filename — is a 1-6 character name whose characters must be A-Z or 0-9, is the filename/extension separator.

ext — is a 1-3 character name whose characters must be A-Z or 0-9. The standard extensions under the Diagnostic Monitor are of the form:

ESD or Txx

ESD is an E&S Diagnostic Monitor system file. These files are priviledged and have restricted access.

T is the type of file:

T=A for ASCII source.

T=L for Load Absolute format.

T=M for MAPSIM statements.
(to be interpreted by MAPSIM.)

T=P for PICSYS statements
(to be interpreted by PICSYS.)

T=S	for Save image binary format. (loadable by Diagnostic Monitor.)
xx	is the version number of the file (01-99).

UPDATE expects file names of this form to be specified where a filename is input. UPDATE also allows an asterisk to be used to replace a file name and/or extension. The use of the asterisk in a file specification means "all". For example, "*.S01" means all files with the extension ".S01" or "PSD100.*" means all files with the file name "PSD100" or "*.*" means all files.

3.2 UPDATE OPERATIONS

A user specifies to UPDATE an operation to be performed by entering a "switch option" either alone or in conjunction with one or several filenames. These operations include transferring files in ASCII or Binary modes and initializing, updating and listing directories of devices. The switches used to perform these operations are detailed in following sections and summarized in Appendix C.

3.2.1 UPDATING FILES

A file may be updated on a device by deleting the file if it already exists and copying the new file from a specified device to the device on which the file is to be updated.

DELETING FILES

A file may be deleted from a device by entering a command of the following form:

```
>dev:filename.ext/D(CR)
```

For example, if the file PSD001.P01 were to be deleted from the system device, the following command would be entered:

```
>PSD001.P01/D(CR)
```

If all files named PSD001 were to be deleted from DT1 the following command would be entered:

```
>DT1:PSD001.*/D(CR)
```

COPYING FILES

A new file may be copied onto a device by entering a command of the following form:

```
>dev:filename.ext=dev:filename.ext/A(CR)  (ASCII format)
```

or

```
>dev:filename.ext=dev:filename.ext/B(CR)  (Binary format)
```

For example, if the file PSP001.A01 were to be copied onto the system device from the high speed paper tape reader, the following command would be entered:

```
>PSP001.A01=PR:/A(CR)
```

If the file PSP001.S01 were to be copied onto DT1 from the system device, the following command would be entered:

```
>DT1:PSP001.S01=PSP001.S01/B(CR)
```

If all files were to be copied from DT0 to DT1 the following command would be entered:

```
>DT1:*.*=DT0:*.*/I/W(CR)
```
[1]

3.2.2 DEVICE DIRECTORIES

INITIALIZING DIRECTORIES

A directory of a given device is initialized by entering a command of the following form:

```
>dev:/Z(CR)
```

UPDATE will respond with the following message:

```
dev:  DO YOU REALLY WANT IT ZEROED?
```

If the user wishes the device to zeroed, a "Y(CR)" should be entered to initiate the zeroing of the device directory.

LISTING DIRECTORIES

A directory of a given device may be listed on the console terminal by entering a command of the following form:

```
>dev:/L(CR)
```

UPDATE will respond by listing the directory of the specified device on the console terminal. The directory will be of the form:

```
filename.ext   #of blocks   creation date
```

for each file in the directory.

---

[1]The /I switch indicates that the files are to be copied Individually. The /W switch indicates that E&S Diagnostic system files (those with extention .ESD) are to be copied also (With permission).

For example, if the directory of DT1 were to be listed, the following command would be entered:

```
>DT1:/L(CR)
```

If a directory of the system device were required, the following command would be entered:

```
>/L(CR)
```

MAINTAINING DIRECTORIES

When files are deleted from a device directory, the entry for that file becomes unuseable until the directory (and the associated files) have been compressed. For that reason, if several files have been updated (i.e. deleted and a new copy installed) the directory should be compressed to ensure that enough storage remains for new files (this need not be done until a "DEVICE FULL" error has occured). A device directory may be compressed by entering command of the following form:

```
>dev:/C(CR)
```

For example, if the directory of the system device were to be compressed, the following command would be entered:

```
>RK0:/C(CR)
```

The warning message "REBOOT?" will be output to remind the user that he should reboot the Diagnostic Monitor if any system files (those with .ESD extention) have been moved on the system device.

3.2.3 BOOTING A DEVICE

UPDATE provides the ability to boot a device (useful when transfering control from the E&S Diagnostic Monitor to the Standard Operating System). A device may be booted by entering a command of the form:

```
>dev:/O(CR)
```

For example, if DT0 is to be booted, the following command would be entered:

```
>DT0:/O(CR)
```

3.2.4 MAKING A BOOTABLE COPY OF A DISK

The following instructions assume that the disk to be copied is DK0: and the disk to receive the copy is DK1: The "DAT" command should always be executed prior to this disk copying operation.

```
>DK1:/Z(CR)
DK1: DO YOU REALLY WANT IT ZEROED? Y(CR)
>DK1:*.*=DK0:*.*/I/W
(This should take a few minutes)
>/Q
>DK1:DUMMY=ESDMON.ESD/U
```

3.3 ERRORS

The following error messages may be output by UPDATE:

ERROR MESSAGE	MEANING
HUH?	Invalid command or command string parameters.

WHAT?	Invalid operation on file.
SWITCH?	Illegal switch or switches.
NO FILE	Filename not found or incorrectly specified in command string.
DEVICE FULL	No room in device directory for further entries.
READ ERROR	Unrecoverable error reading file.
WRITE ERROR	Unrecoverable error writing device directory.
DIRECTORY READ ERROR	Unrecoverable error reading device directory.
DIRECTORY WRITE ERROR	Unrecoverable error writing device directory.
NO BOOT	Device cannot be booted.
BOOT ERROR	Error on boot.
REBOOT?	Reminder to user to reboot system after /C if necessary.

For all of these errors, ensure that the device that was accessed is on-line, ready and write-enabled if required for the operation. The operation may then be re-typed to verify the source of the problem.

CHAPTER FOUR

4.0 PATCH UTILITY PROGRAM (PATCH)

PATCH is a utility program for the E&S Diagnostic System that privides the ability to make code and data modifications to executable image programs that reside on disk or DECtape.

4.1 USING PATCH

To use PATCH, one must request the Diagnostic Monitor to run it from the system (or other specified) device:

```
*RUN PATCH (CR)
```

When PATCH begins execution, it prints the following on the console terminal:

```
PATCH.Sxx     (xx denotes the current version number)
E&S DIAGNOSTIC UTILITY
FILE NAME--
>
```

The > indicates that PATCH is ready to receive the input of a user file which is to be "patched".

Type in the name of the file which is to be modified in the format:

```
[dev:]filename.ext
```

If the device specification [dev:] is omitted, then the system device is assumed.

When the filename.ext which is to be modified has been properly entered, PATCH prints a > indicating that the file is open and PATCH is waiting for a command to be input.

The operations available using PATCH are detailed in Section 4.2.

4.2 PATCHING A PROGRAM

Once a file has been opened for modification, a word address may be examined by a command of the following form:

address/

At this point PATCH will type out the contents of the location and wait for the user to type in a new location contents as an octal value followed by one of these control characters:

<carriage return>--close the current location by changing its contents (if a new contents was specified), and await more control input.

<line feed>-- close the current location and open the next word.

↑ -- close the current location and open the previous word.

@ -- close the current location and open the word addressed by it.

If the contents of the location that is being examined is not to be changed then one of the above control characters may be entered rather than a value, to leave the contents unchanged.

4.3 EXITING FROM PATCH

To exit from PATCH after a program has been modified (or merely examined), the E command may be entered to close the file being modified and return to the Diagnostic Monitor. If multiple files are to be patched (or examined), the F command may be entered to close the file being patched and request that a new file name be typed in.

4.4 PATCH ERROR MESSAGES

INCORRECT FILE SPEC?	The response to the "FILE NAME--" message was not of the correct form. Try again.
MUST OPEN WORD?	The @ command was typed when a byte location was open.
NO ADDR OPEN?	The <line feed>, ↑ or @ command was typed when no location was open.
NOT IN PROGR BOUNDS?	Tried to open a location beyond the end of the file.
ODD ADDRESS?	Tried to open a word address which was odd.
READ ERROR?	File I/O error in reading.
WRITE ERROR?	File I/O error in writing.

CHAPTER FIVE

5.0 PICTURE SYSTEM 2 Diagnostics and Maintenance

This chapter contains an overview of the diagnostic system available for the various components and peripherals of PICTURE SYSTEM 2.

5.1 PICTURE SYSTEM 2 Diagnostics Overview

PICTURE SYSTEM 2 is a stand-alone computer graphics system which can be interfaced to nearly any mini-computer. To avoid the problems encountered when interfacing to another computer, PICTURE SYSTEM 2 and all its supporting diagnostics have been designed to be independent of any computer. However, the particular computer interface and its associated diagnostics must be redesigned for each new computer that is to host a PICTURE SYSTEM 2. This chapter describes the diagnostics available for the Digital Equipment Corp-oration PDP-11/PICTURE SYSTEM 2 Interface and the various hardware components of PICTURE SYSTEM 2.

5.1.1 PICTURE SYSTEM 2 Diagnostic Philosophy

In designing the PICTURE SYSTEM 2 Diagnostic Package, several items were specified as design criteria. These items, listed below, were kept formost in mind as the individual diagnostics were being implemented:

1. ALL PICTURE SYSTEM 2 Diagnostics are written in a "machine independent" assembly language called MIXIT.

The diagnostics are compiled into source assembly code for the particular type of computer upon which they are to run. This code must then be assembled by the Assembler of the target machine and linked together with a machine dependent Diagnostic I/O package to form the PICTURE SYSTEM diagnostics.

2. Regardless of what computer the diagnostics are run on, they will function in the same manner as on the PDP-11 (except for machine dependent peculiarities such as UNIBUS addresses, which are typically invisible to the diagnostician).

3. All the details of the PS2 Diagnostics are specified by the Engineer responsible for designing the component which a particular diagnostic tests. This ensures that all the data paths, registers, etc. are verified and that the sequence of operations is documented for use in understanding the diagnostic.

4. The operator interface is the same for all diagnostics and provides facilities which allow operator selected phases of a diagnostic to be executed repeatedly. Also, error causing conditions may be looped on. This ability enables cards to be debugged easier because conditions which may be "scoped" are set up.

Of these items, number 4 is perhaps the most important to the operator who must run the diagnostics. Each diagnostic is executed in exactly the same manner.

5.1.2 Diagnostic Operating Procedures

A diagnostic is loaded and executed using the E&S

Diagnostic Monitor by typing a command of the following form.

```
*RUN dev:QSD002.S01

      or

*RUN QSD002
```

Note: The system device is assumed when no device is specified. If no extension (ext.) is specified a file with extension .Sxx is assumed. The file with the highest version number (xx = 01-09) will be found and run.

When a diagnostic is executed it types out on the console terminal a message specifying the name of the test and its function, e.g.,

```
QSD002.S01

PICTURE SYSTEM MEMORY/ADDRESS TEST
```

A prompting character ("%") is then typed and the diagnostic waits for operator response.

The operator responds with one of the following characters, followed by a carriage return:

H for "H"elp. This causes the diagnostic to print out a brief synopsis of the nature of each test to be performed, together with the phase number by which that test may be invoked. The program then types "%" and waits for another operator response.

P for "P"ass Count. The program responds by asking

```
HOW MANY PASSES?
```

The operator responds with an unsigned octal number, followed by a carriage return. This instructs the program that each specified phase of the test is to be executed that many times before proceeding to the next specified phase. Thus, a response of "10" would cause each specified phase (see "D" below) to be executed eight times. After accepting this response the program types "%" and waits for another operator response.

D for "D"o These Phases. The program responds by asking

```
DO WHICH PHASE(S)?
```

The operator's response is a string of phase numbers, separated by commas, and followed by a carriage return. This list specifies which phases of the test are to be executed. If this option is not specified, all phases are executed. A typical operator dialogue for this option follows (operator responses are underlined).

```
%D
DO WHICH PHASE(S)?
1,3,5-12,14
%
```

This would specify that execution was to consist of Phases 1, 3, 5 through 12 and 14 (octal).

After accepting this response the program types % and waits for another operator response. Selected phases are always executed in ascending numerical sequence.

X for e"X"ecute. This directs the tests to actually begin. The program types:

```
RUNNING
```

and begins the execution of the selected diagnostic phases.

L for "L"oop on Error. During the course of some previous running of this diagnostic, an error condition may have been detected, and an error message consequently typed out. A typical message might appear as follows:

6: ZERO/ONE ERR; DATA SENT=1777 DATA RECD =1773

The "6" at the beginning of the message is the error Message Number. The "L" option allows the operator to single out an error message (or set of messages) such that, if the associated error is encountered, the program will report the error and then enter an infinite loop, allowing the error conditions to be examined with a scope. A typical dialogue follows:

```
%L
LOOP ON ERROR.  WHICH MESSAGE(S)?
3,6-10
RUNNING
```

The above would cause Message 3, 6, 7 or 10, if encountered, to cause infinite loops. Note that Message Numbers do not necessarily correspond to test phase numbers. The operator may elect to loop indefinitely on the first error encountered, regardless of its number. This he does by entering "-1" for his response. Following the operator's response, the program proceeds as with Option "X" above.

C for "C"ontinue. This option proceeds exactly as Option "L" above. When the test phases encounter one of the selected errors, the message is typed, and the program loops as above -- but only as long as the failure remains. Should the failure condition vanish, the test will type:

CONTINUING

M for "M"odify. Allows the user to examine or change the device and interrupt addresses for the PICTURE SYSTEM 2 and the host computer. These addresses are stored in two tables, one containing the PICTURE SYSTEM addresses, and one containing the host computer's address. Tables H-1 and H-2 list the contents of these two tables.

The program responds to the "M" command by typing

```
MODIFY
*
```

where "*" is the prompt character used while in the modify mode. Four types of table commands are possible - examine parts of the PICTURE SYSTEM table, examine parts of the host computer table, or change a section of either table. Several commands, seperated by semicolons, may appear on one line. Specifically, the commands are:

(1)	$Pn_1=d_1,d_2,\ldots d_m$	(change PICTURE SYSTEM table)
(2)	Pn_1 or $Pn_1-n_2,n_3-n_4,\ldots$	(examine PICTURE SYSTEM table)
(3)	$Hn_1=d_1,d_2,\ldots,d_m$	(change host computer table)
(4)	Hn_1 or $Hn_1-n_2,n_3-n_4,\ldots$	(examine host computer table)

where P - indicates PICTURE SYSTEM table.
H - indicates host computer table.
n_i - is a particular entry number in the range indicated by tables 5-1 and 5-2.
d_i - is the particular value to be stored in the current entry of the table indicated. The value d_1 will be stored in entry n_1, d_2 will be stored in the following entry, and so on, up to entry $n_1 + m$ which will be changed to contain d_m.

The sample command string:

```
P3=3,4,5;P3-5;H1=2;H2=1;H1-2
```

will produce the output

```
3
4
5
2
1
*
```

where "*" is the prompt for another Modify command.

The command "E" returns control to the main program, which will respond with the prompt "%".

There are four error messages in Modify: invalid command, entry out of bounds for table, invalid separator, and attempt to read from a higher to lower entry. The error messages are:

ERR1 X

Meaning: the character "X" was interpreted as a command and not found in the list of acceptable commands (namely P, H, and E). The rest of the command string is ignored.

ERR2 X

Meaning: the entry number "X" was found to be too high or too low for the table currently being accessed. In an examine command, part of this command may already have been executed. In a change command, all modifications up to this point will already have taken place.

ERR3 X

Meaning: the character "X" was interpreted as a separator and not found in the list of acceptable separators (namely =;-,). In an examine command, part of this command may already have been executed. In a change command, all modifications up to this point will already have taken place.

ERR4

Meaning: the user typed in something on the order of "P5-3" instead of "P3-5". The first entry needs to be lower than the second. Can only occur in response to an examine command.

Once a diagnostic begins the execution, testing will proceed according to the constraints set up by the operator, (i.e., Loop on error, etc.) as each diagnostic phase completes its specific number of passes,

PHASE n DONE

is typed, and control proceeds to the next-specified phase. When all selected phases have completed execution, the program types a message such as:

MEMORY TESTS COMPLETE

and either waits for another operator response or returns to the E&S Diagnostic Monitor, dependent upon the particular diagnostic. When running under the E&S Diagnostic Monitor, the operator may type two Control C's at any time to abort the execution of the diagnostic.

5.1.3 Modifying PICTURE SYSTEM 2 Device and Interrupt Addresses

All PICTURE SYSTEM 2 diagnostics are distributed to execute on a standard PICTURE SYSTEM 2/PDP-11 configuration. The standard configuration is:

PICTURE SYSTEM 2/PDP-11 Interface (195131)
UNIBUS ADDRESSES: 167660 - 167676
Interrupt Vectors: 340 - 356

PICTURE SYSTEM 2 Devices:
Real Time Clock (195161) SCB:177744
Picture Generator (195219) SCB:177740
PG Passive Input Port (195219) SCB:177775
Refresh Controller (195151) SCB:177730
Picture Processor (195121) SCB:177750
MAP Passive Input Port (195121) SCB:177777
MAP Passive Output Port (195121) SCB:177776
DMA PSA (195106) SCB:177747
DMA Passive Input Port (195106) SCB:177770
Data Tablet #1 (195181) SCB:177664
Keyboard #1 (195171) SCB:177607
Function Switches #1 (195191) SCB:177626

These device address assignments correspond to the standard switch and jumper positions specified by Figure 5-1. If a diagnostic is to be run on a system with non-standard device assignments, then the device assignment must be reassigned using the "M"odify command available in all diagnostics. The Modify command allows the device assignment tables to be examined and modified. It should be noted that if the diagnostic doesn't reference a table entry it need not be modified. Thus, for example, if the Picture Generator is at a non-standard address, table assignment for the Picture Generator need not be modified for the PS Memory Diagnostic which doesn't use the Picture Generator.

EVANS & SUTHERLAND
COMPUTER CORPORATION
SALT LAKE CITY,
UTAH 84112

SWITCH AND JUMPER POSITIONS

SYSTEM NAME _____ NUMBER _____ S/N _____

SWITCH: 1=OFF (OPEN) Ø=ON (CLOSED)
JUMPER: 1 = JUMP. REMOVED Ø = JUMP. INSTALLED

SLOT NO.	CARD NO.	IC LOC.	1	2	3	4	5	6	7	8	9	IC LOC.	1	2	3	4	5	6	7	8	9	S/N
DEC		11	1	0	0	0	1	1	1	1		20	1	1	1	1	1	0	1	1		
CAB		15	0	0	1	0	0	1	0	1	1	16	0	1	1	1	1	1	1	0	1	
SPC		24	1	1	1	1	1	1	1	0	1	25	1	1	0	1	1	1	1	1	1	
SLOT	195131-100											125	1	1	1	1	1	1	1	1	1	
20	195108-100	27	1	1	1	0	0	1	1	1		29	1	1	1	1	1	0	0	0		
52	195121-100	12	1	1	1	0	1	1	1	1												
49	195151-100	13	1	1	0	1	1	1	1	1												
19	195161-100	72	1	1	1	0	0	1	0	1												
42	195219-100	10	1	1	1	0	0	0	1	1		11	1	1	1	1	1	1	0	1		
45	195222-100	59	1	1																		
4-18	195171-100 1st	23	1	1	0	0	1	1	1			41	1	0	0	0	0	1	1	1		
		40	0	0	0	0	0	0	0	0												
4-18	195181-100 1st	42	0	0	0	0	1	1	1	0		63	0	1	1	0	1	1	0	1		
4-18	195191-100 1st	31	1	1	1	1	1	1	1	0		42	1	1	0	0	1	0	1	1		
4-18	195201-100 1st	U35	1	0		U90		0	1	0	0	U33	0	1	0	0	0	1	1	1		
4-18	195273-100 1st	54	0	0		2nd EXISTS	U54	=	1	0		74	0	1	1	1	0	1	0	0		
10	195175-100 1st	24	0	1	0	1	0	1	0	1	0	BAUD 300										
11	195176-100 1st	64	1	1	1	0	0	0	1	0	1											
4-18	195261-100 1st	10	1	0	0	1	0	0	1	0		44	1	1	0	1	1	1	1	0		

5-013-0

4-18	195171-100 2nd	23	1	1	0	0	1	1	0	1		41	1	0	0	0	0	1	1	0		
		40	0	0	0	0	0	0	0	0												
4-18	195171-100 3rd	23	1	1	0	0	1	0	1	1		41	1	0	0	0	0	1	0	1		
		40	0	0	0	0	0	0	0	0												
4-18	195171-100 4th	23	1	1	0	0	0	1	1	1		41	1	0	0	0	0	1	0	0		
		40	0	0	0	0	0	0	0	0												
4-18	195181-100 2nd	42	0	0	0	0	1	1	0	1		63	0	1	1	0	1	1	0	0		
4-18	195181-100 3rd	42	0	0	0	0	1	0	1	1		63	0	1	1	0	1	0	1	1		
4-18	195181-100 4th	42	0	0	0	0	0	1	1	1		63	0	1	1	0	1	0	1	0		
4-18	195191-100 2nd	31	1	1	1	1	1	1	0	1		42	1	1	0	0	1	0	1	0		
4-18	195191-100 3rd	31	1	1	1	1	1	0	1	1		42	1	1	0	0	1	0	0	1		
4-18	195191-100 4th	31	1	1	1	1	0	1	1	1		42	1	1	0	0	1	0	0	0		
4-18	195201-100 2nd	U35	1	0		U90		0	1	0	1	U33	0	1	0	0	0	1	1	1		
4-18	195273-100 2nd	54	0	1								74	1	0	1	1	0	1	0	1		
6	195175-100 2nd	24	0	1	0	1	0	1	0	1	1	BAUD 300										
7	195176-100 2nd	64																				
4-18	195261-100 2nd	10	1	0	0	1	1	0	1	0		44	0	1	1	1	1	0	1	1		
12	195177-100 1st	10	0	1	0	1	1	1	0	1												
8	195177-100 2nd	10	0	1	1	1	1	1	0	1												

5-013-0

Figure 5-1

PICTURE SYSTEM II Standard Switch Assignments

The only device assignment that must be modified for all diagnostics is the PS-2/PDP-11 UNIBUS address. If the 195131 is DIP switch selected to any UNIBUS address other than 167660 then no diagnostic will execute properly (in fact an illegal memory trap - TRAP to 4 - will probably occur). The standard interrupt vector, 340, need only be modified for the PS-2 interrupt diagnostic - QSD100.

There are two tables that the modify command allows to be examined and/or modified; the HOST Computer Table (H) and the PICTURE SYSTEM Table (P). The contents of the default H and P tables are shown in Appendix H. Note that many entries in the P table are for multiple device systems (i.e., PG #2) and are not used by the standard diagnostics.

For PICTURE SYSTEM installations with non-standard device assignments, rather than using the modify command to alter the device assignments each time a diagnostic is run, the PATCH utility program described in Chapter 4 can be used to permanently modify the device assignment in the diagnostic. To faciliate this, the placement of the P and H tables are the same for all diagnostics:

P Table entry 1 = 1034
H Table entry 1 = 1276

Note: The address of Host Table entry 1 (H) is subject to slight change. In older versions of the diagnostic, this value may be 1166, 1176, or 1210.

For example, to permanently modify the PS-2/PDP-11 UNIBUS address in QSD000, the following command sequence would be used:

```
*RUN PATCH(CR)
PATCH.Sxx (xx denotes the current version number)
E&S Diagnostic Utility
FILE NAME--
>QSD000.S01
>1276/167660 167740(CR)
>E
*
```

5.1.4 Diagnostic Hierarchy

The following grouping of the diagnostics is such that execution of any group generally is meaningful only if all preceding groups execute without error:

1. Processor (e.g. PDP-11) Diagnostics
2. PS-2 Interface Diagnostics QSD000, QSD001
3. PS-2 Memory Diagnostics, including QSD002
4. Interrupt Diagnostic QSD100
5. All other diagnostics, except that Character Generator Diagnostics presuppose proper operation of the Line Generator, and some visual phases of peripherals diagnostics presuppose proper operation of the LG, CG, and Autorefresh.

The diagnostics are written to execute as independently of configuration as possible allowing diagnosis and verification, for example, of PICTURE SYSTEM Memory independently of MAP and Refresh Controller or Line Generator independently of MAP and PICTURE SYSTEM Memory. The individual diagnostic descriptions are included at the end of this chapter.

5.1.5 DESCRIPTIONS OF DIAGNOSTIC PROGRAMS

This section contains detailed descriptions of all standard PS2 Diagnostics.

Name: PS2 DIO INTERFACE DIAGNOSTICS

Functional Description:

Successful execution of this diagnostic indicates that the DIO portion of the CPU/PICTURE SYSTEM interface is operational. It verifies the following:

a. Basic PSBUS operation (REQUEST/GRANT).
b. Correct response to all DIO commands.
c. Correct DIOPSA operation (excluding the validity of the "PSADD" lines on the PSBUS).
d. Correct DIO response to a busy PSBUS device.

Note: This diagnostic does not require PICTURE SYSTEM memory.

The diagnostic consists of seven phases:

1. IOST Write/Read /PSRESET
   Verifies that the Write/Read (W/R) bits of the IOST register can be set, cleared and reset.
2. DMAWC Write/Read/PSRESET
   Verifies that all bits of the DMAWC register can be set, cleared and reset.
3. DMABA Write/Read/PSRESET
   Verifies that all bits of the DMABA register can be set, cleared and reset.
4. DIOPSA Write/Read/PSRESET
   Verifies that all bits of the DIOPSA register can be set, cleared and reset.
5. DIOPSA Increment
   Verifies that the DIOPSA register increments after DIO write and read operations when PSAHOLD is not asserted.
6. DIOPSA Non-Increment
   Verifies that the DIOPSA register does not increment after DIO write and read operations when PSAHOLD is asserted.

7. Device Busy/DIORESET/DIOPSA Passive Load

Verifies that the DIO will not access a busy Passive Port. Verifies that DIORESET will release the DIO from a "hung condition". Verifies that the DIOPSA can be loaded with Passive Port addresses (177770 - 177777).

Program Description:

Phase 1: Verifies that the W/R bits of the IOST can be set, cleared and reset. Program steps are:

a. PSRESET, check for a reset condition.
b. If the condition is not satisfied, the following message is output.

1B: IOST PSRESET ERROR; DATA EXP=XXXXXX DATA REC=XXXXXX

c. Set W/R bits, check result.
d. Clear W/R bits, check result.
e. Repeat steps (a), (c) & (d) several times.
f. Check W/R bits with random data.
g. If step (c), (d) or (f) fails, the following message is output.

1A: IOST W/R ERROR; DATA EXP=XXXXXX DATA REC=XXXXXX

h. Stop - phase complete.

Phase 2: Verifies that all bits of the DMAWC can be set, cleared and reset. Program steps are:

a. Check all bits with data = 0 thru 177777.
b. Check all bits with random data.
c. If step (a) or (b) fails the following message is output.

2A: DMAWC W/R ERROR; DATA EXP=XXXXXX DATA REC=XXXXXX

d. Repeat step (b) several times.
e. Set all bits, PSRESET, check for all zero.
f. If step (e) fails, the following message is output.

2B: DMAWC PSRESET ERROR; DATA EXP=0 DATA REC=XXXXXX

g. Repeat step (e) several times.
h. Stop - phase complete.

Phase 3: Verifies that all bits of the DMABA can be set, cleared and reset. Program steps are:

a. Check all bits with data = 0 thru 177776 (even numbers).
b. Check all bits with random data (even numbers).
c. If step (a) or (b) fails, the following message is output.

3A: DMABA W/R ERROR; DATA EXP=XXXXXX DATA REC=XXXXXX

d. Repeat step (b) several times.
e. Set all bits, PSRESET, check for all zero.
f. If step (e) fails, the following message is output.

3B: DMABA, PSRESET ERROR; DATA EXP=0 DATA REC=XXXXXX

g. Repeat step (e) several times.
h. Stop - phase complete.

Phase 4: Verifies that all bits of the DIOPSA can be set, cleared and reset. Program steps are:

a. Check all bits with data = 0 thru 177767.
b. Check all bits with random data (except 177770-177777).
c. If step (a) or (b) fails, the following message is output.

4A: DIOPSA W/R ERROR; DATA EXP=XXXXXX DATA REC=XXXXXX

d. Repeat step (b) several times.
e. Set high 8 bits, PSRESET, check for all zero.
f. Set low 8 bits, PSRESET, check for all zero.
g. If step (e) or (f) fails, the following message is output.

4B: DIOPSA PSRESET ERROR; DATA EXP=0 DATA REC=XXXXXX

h. Repeat steps (e) and (f) several times.
i. Stop - phase complete.

Phase 5: Verifies that the DIOPSA increments after DIO write and read operations when it does not contain a Passive Port address and PSAHOLD is not asserted. Program steps are:

a. Do a DIO PSBUS read - PSAHOLD=0 (even addresses).
b. Check that DIOPSA did increment.
c. If step (b) fails, the following message is output.

5A: DIOPSA INCREMENT AFTER READ ERROR; DATA EXP=XXXXXX DATA REC=XXXXXX

d. Do a DIO PSBUS write - PSAHOLD=0 (odd addresses).
e. Check that DIOPSA did increment.
f. If step (e) fails, the following message is output.

5B: DIOPSA INCREMENT AFTER WRITE ERROR; DATA EXP=XXXXXX DATA REC=XXXXXX

g. Repeat steps (a), (b), (d) and (e) for addresses 0 thru 177767.
h. Stop - phase complete.

Phase 6: Verifies that the DIOPSA does not increment after DIO write and read operations when it does not contain a Passive Port address and PSAHOLD is asserted. Program steps are:

a. Do a DIO PSBUS read - PSAHOLD=1 (even addresses).
b. Check that DIOPSA did not increment.
c. If step (b) fails, the following message is output.

6A: DIOPSA HOLD AFTER READ ERROR; DATA EXP=XXXXXX DATA REC=XXXXXX

d. Do a DIO PSBUS write - PSAHOLD=1 (odd addresses).
e. Check that DIOPSA did not increment.
f. If step (e) fails, the following message is output.

6B: DIOPSA HOLD AFTER WRITE ERROR; DATA EXP=XXXXXX DATA REC=XXXXXX

g. Repeat steps (a), (b), (d) and (e) for several random addresses (except 177770-177777).

Phase 7: Verifies that the DIO will not access a busy device; that DIORESET will release the DIO from a "hung condition"; that the DIOPSA can be loaded with Passive Port addresses (177770-177777). Program steps are:

a. PSRESET, DMAPIP should be busy.
b. Do a DIO PSBUS read of DMAPIP.
c. Timeout, check that DIOREADY is not set (DIO should be "hung" because DMAPIP was busy).
d. If step (c) fails, the following message is output.

7A: ERROR; DIOREADY SET AFTER READ BUSY DMAPIP ATTEMPT

e. DIORESET, timeout, check that DIOREADY is set.
f. If step (e) fails, the following message is output.

7B: ERROR; DIOREADY CLR AFTER DIORESET

g. Load DIOPSA with a Passive Port address.

h. DIORESET in case the DIO is "hung".
i. Check that DIOPSA contains the Passive Port address.
j. If step (i) fails, the following message is output.

7C: DIOPSA PASSIVE W/R ERROR; DATA=XXXXXX DATA REC=XXXXXX

k. Repeat steps (g) thru (i) several times with data = 177770-177777.
l. Stop - phase complete.

Error Analysis:

Error #	Prognosis
1A:	This error indicates that one or more of the W/R bits of the IOST register (located on the 195131-100 card) could not be either written or read correctly.
1B:	This error indicates that the IOST register (located on the 195131-100 card) could not be reset to its initial state (100200 for the PDP-11 interface) by PSRESET.
2A:	This error indicates that one or more of the bits of the DMAWC register (located on the 195131-100 card) could not be either written or read correctly.
2B:	This error indicates that the DMAWC register (located on the 195131-100 card) could not be reset to zero by PSRESET.

Error #	Prognosis
3A:	This error indicates that one or more bits of the DMABA register (located on the 195131-100 card) could not be either written or read correctly.
3B:	This error indicates that the DMABA register (located on the 195131-100 card) could not be reset to zero by PSRESET.
4A:	This error indicates that one or more bits of the DIOPSA regsiter (located on the 195105-100 card) could not be either written or read correctly. The problem could be the data path to the register, register control logic, the register itself, or the data path from the register.
4B:	This error indicates that the DIOPSA register (located on the 195105-100 card) could not be reset to zero by PSRESET.
5A:	This error indicates that the DIOPSA register (located on the 195105-100 card) did not increment correctly after a DIO PSBUS write operation with PSAHOLD not asserted.
5B:	This error indicates that the DIOPSA register (located on the 195105-100 card) did not increment correctly after a DIO PSBUS read operation with PSAHOLD not asserted.
6A:	This error indicates that the DIOPSA register (located on the 195105-100 card) did not retain the contents it had prior to a DIO PSBUS read operation with PSAHOLD asserted.

Error #	Prognosis
6B:	This error indicates that the DIOPSA register (located on the 195105-100 card) did not retain the contents it had prior to a DIO PSBUS write operation with PSAHOLD asserted.
7A:	This error indicates that the DIOREADY bit (located in the IOST register on the 195131-100 card) was set after the DIO tried to access the DMAPIP which should have been busy due to PSRESET. Access of this busy Passive Port should have caused the DIO to "hang" (i.e., DIOREADY should have been clear).
7B:	This error indicates that a DIORESET would not release the "hung" condition the DIO was in due to trying to access the busy DMAPIP. (i.e., cause the DIOREADY bit located in the IOST register on the 195131-100 card to be set).
7C:	This error indicates that the DIOPSA register (located on the 195105-100 card), could not be either written or read correctly when loaded with the address of a Passive Port (i.e., DATA = 177770-177777).

Name: PS2 DMA INTERFACE DIAGNOSTICS

Functional Description:

Successful execution of this diagnostic indicates that the DMA portion of the CPU/PICTURE SYSTEM interface is operational. It verifies the following:

a. All DMA modes of operation (Active Output, Active Input and Passive Input).
b. Correct DMAPSA operation (excluding the validity of the "PSADD" lines on the PSBUS).
c. Correct DMA response to a busy PSBUS device.
d. Correct DMA Word Count Register (DMAWC) operation.
e. Correct DMA Bus Address Register (DMABA) operation including extended address bits.
f. Correct DMA response to a non-existent PDP-11 memory location access (NEXMEM).
g. Execution of simultaneous DIO & DMA block data transfers to and from PICTURE SYSTEM memory.

Note: This diagnostic assumes successful exectuion of QSD000. Phases 1 thru 12 do not require PICTURE SYSTEM memory. Phases 13 & 14 require the first 1000 words of PICTURE SYSTEM Memory and assume they can be written and read correctly via the DIO.

The diagnostic consists of twelve phases:

1. DMAPSA Write/Read/PSRESET
   Verifies that all bits of the DMAPSA register can be set, cleared and reset.
2. DMAPSA Increment
   Verifies that the DMAPSA register will increment after DMA Active Output/Input operations if it does not contain a Passive Port address prior to the operation.

3. Device Busy/DMARESET/DMAPSA Non-Increment

   Verifies that the DMA will not access a busy Passive Port. Verifies that DMARESET will release the DMA from a "hung condition". Verifies that the DMAPSA will not increment after DMA Active Output/Input operations if it contains a Passive Port address prior to the operation.

4. DMA Active Output to DMAPSA

   Verifies the DMA Active Output mode of operation.

5. DMA Active Input from DMAPSA

   Verifies the DMA Active Input mode of operation.

6. DMA Passive Input from DIO

   Verifies the DMA Passive Input mode of operation.

7. DMA Active Input to NEXMEM

   Verifies that the DMA responds correctly when accessing a non-existent PDP-11 memory location.

10_8. DMAWC Increment

   Verifies that the DMAWC register increments correctly after each DMA data transfer.

11_8. DMABA Increment

   Verifies that the DMABA register increments correctly after each DMA data transfer.

12_8. XBA(17-16) Increment

   Verifies that the Extended Bus Address bits increment each time the DMABA overflows (177776→0).

13_8. DMA Block Transfers Out/In

   Verifes that the DMA can transfer blocks of data to (Active Output) and from (Active Input) PICTURE SYSTEM Memory.

14_8. Simultaneous DIO & DMA Block Transfers

   Verifies that the DIO can transfer a block of data to or from one area of PICTURE SYSTEM Memory while the DMA is in the process of transfering a block of data to or from another area of PICTURE SYSTEM Memory.

Program Description:

Phase 1: Verifies that all bits of the DMAPSA can be set, cleared and reset. Program steps are:

a. Check all bits with data = 0 thru 177777.
b. Check all bits with random data.
c. If step (a) or (b) fails, the following message is output.

1A: DMAPSA W/R ERROR; DATA EXP=XXXXXX DATA REC=XXXXXX

d. Repeat step (b) several times.
e. Set all bits, PSRESET, check for all zero.
f. If step (e) fails, the following message is output.

1B: DMAPSA PSRESET ERROR; DATA EXP=0 DATA REC=XXXXXX

g. Repeat step (e) several times.
h. Stop - phase complete.

Phase 2: Verifies that the DMAPSA will increment after DMA Active Output/Input operations if it does not contain a Passive Port address prior to the operation. Program steps are:

a. Initiate a single DMA Active Output transfer (DMAWC=177777; DMABA=address of a PDP-11 memory location containing zero; Set GO in the IOST).
b. Verify that the DMAPSA did increment.
c. If step (b) fails, the following message is output.

2: DMAPSA INCREMENT ERROR; DATA EXP=XXXXXX DATA REC=XXXXXX

d. Repeat steps (a) and (b) with DMAPSA = 0 thru 177776 (except for DMAPSA PSBUS address).
e. Stop - phase complete

Phase 3: Verifies that the DMA will not access a busy Passive Port; that DMARESET will release the DMA from a "hung condition"; that the DMAPSA will not increment after DMA Active Output/Input operations when it contains a Passive Port address prior to the operation. Program steps are:

a. PSRESET, DMAPIP should be busy.
b. Initiate a single DMA Active Output transfer to the DMAPIP (DMAWC=177777; DMABA=address of a PDP-11 memory location containing zero; Set GO in the IOST).
c. Timeout, verify that DMAREADY is not set (the DMA should be "hung" since it accessed a busy Passive Port).
d. If step (c) fails, the following message is output.

3A: ERROR; DMAREADY SET AFTER BUSY DMAPIP ACCESS

e. DMARESET, timeout, verify that DMAREADY is set.
f. If step (e) fails, the following message is output.

3B: ERROR; DMAREADY CLR AFTER DMARESET

g. Initiate a single DMA Active Output transfer (DMAWC=177777; DMABA=address of a PDP-11 memory location containing zero; Set GO in the IOST).
h. DMARESET in case the DMA is "hung".
i. Verify that the DMAPSA did not increment.
j. If step (i) fails, the following message is output.

3C: DMAPSA NON-INCREMENT ERROR; DATA EXP=XXXXXX DATA REC=XXXXXX

k. Repeat steps (g) thru (i) several times with DMAPSA=177770-177777.
l. Stop - phase complete.

Phase 4: Verifies the DMA Active Output mode of operation by doing single DMA transfers to the DMAPSA. Program steps are:

a. Initiate a single DMA Active Output transfer to the DMAPSA (DMAWC=177777; DMABA=address of a PDP-11 memory location containing zero; Set GO in the IOST).
b. Check the DMAPSA for correct contents.
c. If step (c) fails, the following message is output.

4: DMA ACTIVE OUTPUT TO DMAPSA ERROR; DATA EXP=XXXXXX DATA REC=XXXXXX

d. Repeat steps (a) and (b) several times with data=0 thru 177776 (even numbers).
e. Repeat steps (a) and (b) several times with random data (even numbers).
f. Stop - phase complete.

Phase 5: Verifies the DMA Active Input mode of operation by doing a single DMA transfer of the contents of the DMAPSA to CPU memory. Program steps are:

a. Clear a PDP-11 memory location.
b. Load the DMAPSA with its PSBUS address.
c. Initiate a single DMA Active Input transfer to a PDP-11 memory location. (DMAWC=177777; DMABA=PDP-11 memory location address; Set DMAIN and GO bits in the IOST).
d. Check the PDP-11 memory location for DMAPSA PSBUS address.
e. If step (d) fails, the following message is output.

5: DMA ACTIVE INPUT FROM DMAPSA ERROR; DATA EXP=XXXXXX DATA REC=XXXXXX

f. Repeat steps (a), (b), (c) and (d) several times.
g. Stop - phase complete.

Phase 6: Verifies the DMA Passive Input Mode of operation by directing DIO data transfers to the DMA Passive Input Port (DMAPIP). Program steps are:

a. Initiate a single DMA Passive Input transfer (DMAWC=177777; DMABA=PDP-11 memory location address; set PASSIVE, DMAIN and GO bits in the IOST).
b. Do a DIO data transfer to the DMAPIP.
c. Check the PDP-11 memory location for correct data.
d. If step (c) fails, the following message is output.

6: DMA PASSIVE INPUT FROM DIO ERROR; DATA EXP=XXXXXX DATA REC=XXXXXX

e. Repeat steps (a), (b) and (c) several times with random data.
f. Stop - phase complete.

Phase 7: Verifies that the DMA will not "hang" when a data transfer causes a nonexistent PDP-11 memory access. Program steps are:

a. Initiate a two word DMA Active Input transfer to a non-existent PDP-11 memory location. (DMAWC=177776; DMABA=173000; set DMAIN, XBA (17-16) and GO bits in the IOST).
b. Check the DMAWC, DMABA and IOST for correct contents.
c. If step (b) fails, the following message is output.

7: DMA ACTIVE INPUT TO NEXMEM ERROR;
DMAWC: DATA EXP=177777 DATA REC=XXXXXX
DMABA: DATA EXP=173002 DATA REC=XXXXXX
IOST: DATA EXP=140266 DATA REC=XXXXXX

d. Repeat steps (a) and (b) several times.
e. Stop - phase complete.

Phase 10: Verifies that the DMAWC register increments correctly after each DMA data transfer. Program steps are:

a. Initiate DMA Active Input transfers to a non-existent PDP-11 memory location (DMABA=173000; set DMAIN, XBA (17-16) and GO bits in the IOST).
b. Verify that the DMAWC did increment.
c. If step (b) fails, the following message is output.

10: DMAWC INCREMENT ERROR; DATA EXP=XXXXXX DATA REC=XXXXXX

d. Repeat steps (a) and (b) with DMAWC=0 thru 177777.
e. Stop - phase complete.

Phase 11: Verifies that the DMABA register increments correctly after each DMA data transfer. Program steps are:

a. Initiate a single DMA Active Output transfer (DMAWC=177777; set GO in the IOST).
b. Verify that the DMABA did increment.
c. If step (b) fails, the following message is output.

11: DMABA INCREMENT ERROR; DATA EXP=XXXXXX DATA REC=XXXXXX

d. Repeat steps (a) and (b) with DMABA=0 thru 177776.
e. Stop - Phase complete.

Phase 12: Verifies that the Extended Bus Address bits increment each time the DMABA overflows (177776 → 0). Program steps are:

a. Initiate a single DMA Active Output transfer (DMAWC=177777; DMABA=177776; set GO in the IOST).

b. Verify that the XBA bits did increment.
c. If step (b) fails, the following message is output.

12: XBA (17-16) INCREMENT ERROR; DATA EXP=XXXXXX DATA REC=XXXXXX

d. Repeat steps (a) and (b) several times with XBA (17-16) = 00, 01, 10 and 11.
e. Stop - phase complete.

Phase 13: Verifies that the DMA can transfer blocks of data to and from PICTURE SYSTEM memory. Program steps are:

a. Initiate a 500 word DMA Active Output transfer to PICTURE SYSTEM memory locations 0 thru 499 (DMAWC=177014; DMABA=PDP-11 memory "data block out" starting address; set GO in the IOST).
b. When the output transfer completes, initiate a 500 word DMA Active Input transfer from PICTURE SYSTEM memory locations 0 thru 499. (DMAWC=177014; DMABA=PDP-11 memory "data block in" starting address; set DMAIN and GO bits in the IOST).
c. When the input transfer completes, verify that the block of data sent to PICTURE SYSTEM memory is identical to the block of data received from PICTURE SYSTEM memory.
d. If step (c) fails, the following message is output.

NNN; 13: DMA BLOCK TRANSFER ERROR; DATA EXP=XXXXXX DATA REC=XXXXXX

Where: NNN = the octal data transfer number ($0 \leq NNN \leq 763$)

e. Repeat steps (a), (b) and (c) several times with blocks of random data.
f. Stop - phase complete.

Phase 14: Verifies that simultaneous DMA and DIO data transfers can be performed. Program steps are:

a. Initiate a 500 word DMA Active Output transfer to PICTURE SYSTEM memory locations 500 thru 999 (DMAWC=177014; DMABA=PDP-11 memory "data block out" starting address; set GO in the IOST).
b. Without waiting for the DMA transfer to finish, initiate a 500 word DIO transfer to PICTURE SYSTEM memory locations 0 thru 499.
c. When both DMA and DIO transfers are complete, initiate a 500 word DMA Active Input transfer from PICTURE SYSTEM memory locations 0 thru 499 (DMAWC=177014; DMABA=PDP-11 memory "data block in" starting address; set DMAIN and GO bits in the IOST).
d. Without waiting for the DMA transfer to finish, initiate a 500 word DIO transfer from PICTURE SYSTEM memory locations 500 thru 999.
e. When both DMA and DIO transfers are complete, verify that the 1000 words sent to PICTURE SYSTEM memory are identical to the 1000 words received from PICTURE SYSTEM memory.
f. If step (e) fails, the following message is output.

NNNN; 14: DIO OR DMA BLOCK TRANSFER ERROR; DATA EXP=XXXXXX DATA REC=XXXXXX

where: NNNN = the octal data transfer number ($0 \leq NNNN \leq 1747$).

g. Repeat steps (a), (b), (c), (d) and (e) several times with blocks of random data.
h. Stop - phase complete.

Error Analysis:

Error #	Prognosis
1A:	This error indicates that one or more bits of the DMAPSA register (located on the 195106-100 card) could not be either written or read correctly. The problem could be the data path to the register, register control logic, the register itself, or the data path from the register to the PSBUS.
1B:	This error indicates that the DMAPSA register (located on the 195106-100 card) could not be reset to zero by PSRESET.
2:	This error indicates that the DMAPSA register (located on the 195106-100 card) did not increment correctly after a DMA Active Output operation when it did not contain a Passive Port address prior to the operation. The problem may be the DMA Active Output control logic on the 195131-100 card or the logic which generates the *INCDMAPSA signal on the 195106-100 card.
3A:	This error indicates that the DMAREADY bit (located in the IOST register on the 195131-100 card), was set after the DMA tried to access the DMAPIP which should have been busy due to PSRESET. Access of this busy Passive Port should have caused the DMA to "hang" (i.e., DMAREADY should have been clear).

Error #	Prognosis
3B:	This error indicates that a DMARESET would not release the "hung" condition the DMA was in due to trying to access a busy Passive Port (DMAPIP). DMARESET should have caused the DMAREADY bit (located in the IOST register on the 195131-100 card) to be set.
3C:	This error indicates that the DMAPSA register (located on the 195106-100 card) did not retain the Passive Port address it contained prior to a DMA Active Output operation. The *INCDMAPSA (195106-100 card) should not occur.
4:	This error indicates that the DMAPSA register (located on the 195106-100 card) did not contain correct data after a DMA Active Output transfer to the register. Bit zero should always be loaded with zero so the entire register will not be affected by the *INCDMAPSA signal at the end of the transfer. Bit zero is not checked as part of a correct result. The problem is likely to be the DMA data path to the register.
5:	This error indicates that correct data was not received when the DMAPSA register (located on the 195106-100 card) was used as the source of data during a DMA Active Input transfer. The data received should be equal to the DMAPSA PSBUS address. The problem may be the DMA Active Input control logic on the 195131-100 card or the DMA data path from PSBUS to CPU.

Error #	Prognosis
6:	This error indicates that correct data was not received when DIO data transfers were directed to the DMA Passive Input Port (DMAPIP). The problem may be the DMAPIP address response logic which generates AEPASDATI (195106-100 card), control logic which responds to AEPASDATI, or the DMA data path from PSBUS to CPU.
7:	This error indicates that the DMA did not "hang" when a DMA data transfer caused a non-existent PDP-11 memory address access but that either the DMAWC register, DMABA register or IOST register, (all located on the 195131-100 card) contained incorrect data after the transfer. If the DMA does "hang" the UNIBUS waiting for SSYN from a non-existent PDP-11 slave, the phase will never finish. This indicates that the "MSYN timeout" logic on the 195131-100 card is not working.
10:	This error indicates that the DMAWC register (located on the 195131-100 card) did not increment correctly after a DMA Active Input transfer to a non-existent PDP-11 memory location. The first transfer should cause NEXMEM to set DMAREADY (both located in the IOST on the 195131-100 card) independent of the contents of the DMAWC. The problem is likely in the DMA UNIBUS control logic on the 195131-100 card.
11:	This error indicates that the DMABA register (located on the 195131-100 card) did not increment correctly after a DMA Active Output transfer. The problem is likely in the DMA UNIBUS control logic on the 195131-100 card.

Error #	Prognosis
13:	This error indicates that there was a data error during either a DMA block transfer to or from PICTURE SYSTEM memory. If many bits are in error, it is usually an indication of a timing problem within the DMA control logic or data path. Board swapping (195105-100, 195106-100, 195131-100) is the fastest way to solve the problem since it is very hard to loop on a 500 word data transfer.
14:	This error indicates that there was a data error during either simultaneous DIO or DMA data transfer to or from PICTURE SYSTEM memory. It usually indicates an arbitration or timing problem within DIO/DMA control logic or DIO/DMA data path usage. Board swapping (195105-100, 195106-100, 195131-100) is the fastest way to solve the problem if many bits are in error or if no failure pattern can be established.

Identification: QSD002
Date: 11/3/76

Name: PICTURE SYSTEM MEMORY TEST

Functional Description:

This diagnostic is used for testing the Picture System Memory. Since the amount of memory may vary from 16K to 64K words, the program will ask the operator to respond with the amount of memory on the system. (codes are: 1=16K words, 2=32K words, 3=48K words, and 4=64K words). The diagnostic consists of seven phases:

1. Memory Data Path Check.
2. Memory Address/Data Test.
3. Zero/one Test.
4. Random Number Test.
5. Refresh Test.
6. Bit Disturb Ones Test.
7. Bit Disturb Zeroes Test.

Program Description:

Phase 1: This test checks memory request/acknowledge, timing, and data paths. Five data values are checked: 0, 177777, 125252, 052525, and 123456. Each set of memory cards contains 16K words of memory. Therefore to check data paths on all memory cards in the system, the test must be made at addresses: 0, 040000, 100000, and 140000 (0-16K, 16K-32K, 32K-48K, and 48K-64K respectively). Two data paths (B,C) exist on all memory cards. Initially path C (controlled by port 0) is checked. Then the user is asked to ground pin 62 on the backpanel slot of the 195141-100 card, and the test is run again to test path B (controlled by port 1).
If an error occurs, the following message is output.

1: DATA PATH ERR; PORT=X ADDR=XXXXXX DATA SENT=XXXXXX DATA RECD=XXXXXX

Phase 2: This test is an address and data check, with its main purpose to check the address registers of port 0 and port 1.

The test is executed by writing the address value into each memory location, then reading and checking all memory locations. In order to check port 1, the user is asked to ground pin 62 on the backpanel slot of the 195141-100 card.
If an error occurs, the following message is output.

2: ADDRESS ERR; PORT=X ADDR=XXXXXX DATA SENT=XXXXXX DATA RECD=XXXXXX

Phase 3-7: These tests are designed to check the contents of the 4K MOS memory integrated circuits. Since both port 0 and port 1 are common to the integrated circuits, only port 0 needs to be checked in these phases.

Phase 3: Zero/One Test
The memory is set to all ones then checked. All the even locations are then complemented and the memory is checked for alternating zeros and ones. The odd memory locations are then complemented and the memory is checked for all zeros. Once again the even memory locations are complemented and the memory checked for alternating ones and zeros.
If an error occurs, the following message is output.

3: ZERO/ONE ERR; ADDR=XXXXXX DATA SENT=XXXXXX DATA RECD=XXXXXX

Phase 4: Random Number Test
This test loads the memory with a sequence of psuedo-random numbers. After loading the memory, it is checked by generating the same series of psuedo-random numbers.
If an error occurs, the following message is output.

4: RANDOM DATA ERR; ADDR=XXXXXX DATA SENT=XXXXXX DATA RECD=XXXXXX

Phase 5: Refresh Test

The memory is filled with pseudo-random numbers then the program waits for approximately one minute without accessing the memory. The memory is then checked by generating the same series of pseudo-random numbers. If an error occurs, the following message is output.

5: RANDOM DATA ERR; ADDR=XXXXXX DATA SENT=XXXXXX DATA RECD=XXXXXX

Phase 6: Bit Disturb Ones Test

The memory is initially set to all ones. A single location is then complemented N times. This will complement one bit in each of 16 memory chips. The chip row that contains the location under test is then checked for all ones. This is then repeated for all memory locations.

If an error occurs, the following message is output.

6: BIT DISTURB ERR; ADDR=XXXXXX DATA SENT=XXXXXX DATA RECD=XXXXXX

Phase 7: Bit Disturb Zeros Test

This test uses the same procedure as the BIT DISTURB ONES TEST except that the memory is initially set to zero and checked for zero after the complementing.

If an error occurs, the following message is output.

7: BIT DISTURB ERR; ADDR=XXXXXX DATA SENT=XXXXXX DATA RECD=XXXXXX

Error Analysis:

Error #	Prognosis
1.	If many data bits are in error in each word, then control or addressing is the probable cause. The control cards are: 195107-100, 195142-101, and 195141-100. Addressing logic is on the 195141-100 card. If a few (1-4) bits are in error, then data paths on the 195143-101 or 195107-100 cards are the probable cause.

Error #	Prognosis
2.	If many data bits are in error in each word, then the addressing is the probable cause. The addressing logic is on the 195141-100 card. If a few (1-4) bits are in error, then the memory cards (195143-101) are probable cause.
3-7.	All these tests are for checking the 4K MOS integrated circuits. If this test fails, the memory IC's on the 195143-101 cards should be checked. The following table is to be used to cross reference from the error message to the integrated circuit.

Address of Error (ADDR)	Card pair (195143-101) backpanel slots Card A	Card B
0 - 37777	25	26
40000 - 77777	27	28
100000 - 137777	29	30
140000 - 177377	31	32

From the above determine error card A and error card B. Then from the following table, determine which IC position on card A or B caused the error.

w 14 bits error dress	B								A								card data in err
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
777	53	43	33	27	49	39	29	18	53	43	33	27	49	39	29	18	
00-17777	54	44	34	37	50	40	30	28	54	44	34	37	50	40	30	28	
00-27777	55	45	35	47	51	41	31	38	55	45	35	47	51	41	31	38	IC pos.
00-37777	56	46	36	57	52	42	32	48	56	46	36	57	52	42	32	48	

Identification: QSD003
Date: 3/1/77

Name: (MAP) STATUS, MAINTENANCE, AND MATCH TESTS

Functional Description:

This test consists of two phases:

1. Test read/write access and system reset in the Map Status Register (MSR:177753 or SR), Map Maintenance Status Register (MMSR:177754 or MSR) and Map-related bits of the System Interrupt Enable register (SYSIE:177763).
2. Test read/write access to Map Active Output Address (MAOA:177751 or MAROUT) and Map Active Output Limit (MAOL:177750 or MAREND). Also, verify the operation of the MAP Output Stopped Request (MOSTOP REQ) in the System Interrupt Request register (SYSREQ:177762).

Program Description:

Phase 1: Test read/write and reset access in MSR, MMSR, and SYSIE.

a. Issue Map Reset by writing 1 into MSR.
b. Test MSR for the value 120200. If error, then output the message:

1: MAP RESET ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=12

c. Test MMSR for: MAPMNT and MAPHLT set; OUTBUSY, NV, and SV clear; WRDSEL=0. If error, then output the message:

1: MAP RESET ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=16

d. Test SYSIE for: bits 3:0=0. If error then output the message:

1: MAP RESET ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=21

e. Repeat step (f) 8 times using the data sequence 1, 3, 7, 17, 16, 14, 10, 0.

f. Write the next data pattern into SYSIE. Test SYSIE bits 3:0 for expected data. If an error occurs, then output the message:

2: BIT SET/CLR ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=45

g. Test Mask = 163407 (see note below). Repeat step (h) using the data sequence 2000, 3000, 3400, 3404, 3406.

h. Write the next data pattern into MMSR, then test MMSR. If error:

2: BIT SET/CLR ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=57

i. Test Mask = 17577. Repeat step (j) 8 times using the data sequence 2, 6, 16, 36, 76, 176, 576, 1576.

j. Write and test MSR. If error, then output the message:

2: BIT SET/CLR ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=71

k. Test IV and TAKE; write 7576 into MSR, then test MSR for 5576. If error, then output the message:

2: BIT SET/CLR ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=75

Note: Bits set in the "Test Mask" are those bits capable of generating an error; bits not set in the Test Mask are ignored. Specifically, an error occurs only if ((Expected Data) XOR (Received Data)) AND (Test Mask) does not equal 0. The default value of the Test Mask is naturally, 177777.

l. Test HIT and TAKE; write 17576 into MSR, then test MSR for 15576. If error, then output the message:

2: BIT SET/CLR ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=80

m. Repeat step (n) using the data sequence 15574, 15570, 15560, 15540, 15500, 15400, 15000, 14000.

n. Write and test MSR. If error, then output the message:

2: BIT SET/CLR ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=105

o. Test the MSR take bits; repeat step (p) four times with the input data sequence 0, 12000, 2000, 14000, and the expected data sequence 14000, 10000, 0, 0.

p. Write the input data into MSR. Test for expected data. If error, then output the message:

2: BIT SET/CLR ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=115

q. Test Mask = 3407, for MMSR.
r. Repeat step (s) five times, using the data sequence 1406, 406, 6, 3, 0.
s. Write and test MMSR. If error, then output the message:

2: BIT SET/CLR ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=127

t. Test Mask = 17577, for MSR. If MSR not equal 0, then output the following message:

2: BIT SET/CLR ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=134

u. Test Mask = 17, for SYSIE. If SYSIE not equal 0, then:

2: BIT SET/CLR ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=140

v. MMSR←3406. MSR←17576. MSR←17577 (Reset, writing all bits set). Test Mask=177777.

w. Test MSR for 120200. If error:

2: BIT SET/CLR ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=157

x. Test Mask = 163407. Test MMSR for 6. If error:

2: BIT SET/CLR ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=163

y. SYSIE←17. MSR←1 (Map Reset). Test Mask = 17. Test SYSIE for 0. If error:

2: BIT SET/CLR ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=173

z. End Phase 1.

Phase 2:

a. For registers MAOA, MAIA, and MAOL, and the data sequence 0, 177777, 125252, 52525 and 123456: load and verify. If an error occurs, then output the following message:

3: REG SET/CLR ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX =xx (17=MAOA, 22=MAOL, 25=MAIA)

b. Load MAOA and MAOL with unequal values, and clear the interrupt request bit, MOSTOP REQ (in SYSREQ).

c. Load MAOA with a value unequal to MAOL. If MOSTOP REQ now = 1, then output the following message:

4: MATCH FAILURE; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=46

d. Load MAOL with a value unequal to MAOA. If MOSTOP REQ now = 1, then output the following message:

4: MATCH FAILURE; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=52

e. Load MAOA with a value equal to MAOL. If MOSTOPPED REQ still = 0, then output the following message:

4: MATCH FAILURE; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=55

f. Load MAOA with a value unequal to MAOL. If MOSTOPPRED REQ still = 0, then output the following message:

4: MATCH FAILURE; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=61

g. End of test.

Error Analysis:

This test checks the MAP Status Register (MSR), the Map Maintenance Status Register (MMSR), the System Interuupt Enables (SYSIE), and the System Interrupt Requests (SYSREQ). The following tables give the drawing location where each bit of the registers can be found.

MSR

Bit #	Card #	Sheet	Zone #
0	195121	4	B2
1	195121	4	C3 (*HLDUPDT)
2	195121	4	C3 (*HLDRSR)
3	195121	4	C3 (*HIT TEST ENB)
4	195121	4	C3 (*ACTIVEI)
4	195118	5	B4 (*ACTIVEI)
(Bit 4 has two registers containing '*ACTIVEI'. The register on the 195118 card is checked in this test).			
5	195118	5	B4 (*ACTIVEO)
6	195118	5	B4 (*OUTPUTINHIB)
7	195119	3	A1 (MAREQL)
8,9	195117	3	A2 (MODEA,MODEB)
10	195117	2	C3 (PSDATI0)
11	195117	2	B2 (IB)
12	195117	2	B2 (HIT)
13	195119	1	B2 (FEMPTY)
14	195117	4	D3 (FULL)
15	195121	5	C1 (*HOLD)
(also see	195117	3	C4)

MMSR

Bit #	Card #	Sheet #	Zone #
0	195121	4	B2 (*SNGLSTP)
1	195121	4	B3 (MAPHALT)
2	195121	4	B3 (MNT
3-7	Not Used		
8-10	195117	4	C2 (MDSELA, MDSELB, MOSELC)
11	Not Used		
12	195121	5	C1 (*HOLD)
(also see	195117	3	C3 (*HOLD))
13	195117	1	C2 (SV)
14	195117	1	C2 (NV)
15	195118	2	B2 (OUTPUT BUSY)
(also see	195117	3	C4 (OUTPUT BUSY))

SYSIE

Bit #	Card #	Sheet #	Zone #
0	195121	3	A2 (MAPHLTENB)
1	195121	3	A2 (HITENB)
2	195121	3	A3 (JMPENB)
3	195121	3	A3 (MATCHENB)

SYSREQ

Bit #	Card #	Sheet #	Zone #
0	195121	3	B3 (MAPHLTREQ)
1	195121	3	B3 (HITREQ)
2	195121	3	C3 (JMPREQ)
3	195121	3	D3 (MATCHREQ)

Other Registers

Register	Card #	Sheet #	Zone #
MAOA	195119	3	D1-4
MAOL	195119	3	C1-4
MAIA	195119	4	C1-4
comparator	195119	3	B1-4

Error #	Index #	Prognosis
1	12	MSR cannot be reset or read.
1	16	MMSR cannot be reset or read.
1	21	SYSIE cannot be reset or read.
2	45	SYSIE cannot be written and/or read.
2	57	MMSR cannot be written and/or read. (Check all bits except bits 12, 11, 7-3).
2	71	MSR cannot be written and/or read (Check all bits except bits 15, 14, 13, 7).
2	75	'IB' of MSR cannot be written and/or read.
2	80	'HIT' of MSR.
2	105	MSR cannot be written and/or read. (In this test, HIT & IB should remain set).
2	115	'HIT' and 'IB' cannot be cleared in this test.
2	127	MMSR bits 10, 9, 8, 2, 1, 0 cannot be written and/or read.
2	134	MSR is checked for zeroes. (previous parts of this test should leave it in the zero state. - bit 10 is not checked).
2	140	Bits 0, 12, 3 of SYSIE are checked for zeroes.
2	157	MSR cannot be reset.
2	163	MMSR cannot be reset.
2	173	SYSIE cannot be reset.

Error #	Index #	Prognosis
3	17	The MAOA register cannot be written and/or read.
3	22	The MAOL register cannot be written and/or read.
3	25	The MAIA regsiter cannot be written and/or read.
4	46	The MOSTOP REQ bit was set to a 1 in error. Since MAOA was set to a different value than MAOL, the comparator should not have sensed equality and set MOSTOP REQ. (The hardware name of MOSTOP REQ is MATCH REQ).
4	52	The MOSTOP REQ bit was set to a 1 in error. Since MAOL was set to a different value than MAOA, the comparator should not have sensed equality and set MOSTOP REQ. (The hardware name of MOSTOP REQ is MATCH REQ).
4	55	The MOSTOP REQ bit was not set to a 1. Since MAOA was loaded with the same value as MAOL, the comparator should have sensed equality and set MOSTOP REQ.
4	61	MAOA was set to a value different than MAOL. However since MOSTOP REQ was already set to 1 (from previous test), MOSTOP REQ should remain a 1. This error indicates it somehow was cleared.

Name: PROM/RAM & DOIT REGISTER TEST

Functional Description:

This diagnostic is used to test the PROM/RAM address register (PRMADR) and the DOIT register. The diagnostic consists of four phases:

1. DOIT Address Test
   Checks PRMADR with fixed data patterns.
2. Address Incrementation
   Checks WRDSEL for incrementing each time the MAP BBUS is read, and checks PRMADR for incrementing each 8th time the MAP BBUS is read.
3. DOIT Bits
   The 96 control (DOIT) bits are checked with fixed data patterns by writing and reading the MAP BBUS.
4. PROM/RAM Content
   The contents of the control PROMS are checked by strobing each of the 256 locations into the DOIT register and checking the MAP BBUS with the standard control store contents contained within the diagnostic.

Program Description:

Phase 1: With the MAPMNT bit of the MMSR register set to 1, the PROM/RAM Address field of the MMPAR register is checked with the following data patterns:
0,377,252,125,456.
If an error occurs the following message is output.

1: DOIT ADDR ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=23

Phase 2: Check the incrementation of the PROM/RAM Address field (PRMADR) of the MMPAR Register and the control store word select field (WRDSEL) of the MMSR register.

a. Set the MAPMNT bit of the MMSR register to 1.
b. Set WRDSEL to 7.
c. Set the PRMADR to -1.
d. Read MAP BBUS.
e. Check to see if register incremented. If not the following message is output.

2: PROM/RAM ADDR ERR; ADDR=XXXXXX EXPT=XXXXXX RECD=XXXXXX INDEX=or 37 41

f. Repeat steps d and e 2048 times.
g. Stop - phase complete.

Phase 3: Using the PROM/RAM Address field (PRMADR) of the MMPAR register and the word select field (WRDSEL) of the MMSR register each of the twelve bytes (8 bits per byte) of the DOIT register is checked with the following patterns:

0,377,252,125,456.

If an error occurs the following message is output.

3: DOIT LOAD ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=40 WSF=n

where n is the contents of the WRDSEL field.

Phase 4: The micro-program of the control store is checked with the standard microcode contained in a table within this diagnostic.

a. Set the MAPMNT bit of the MMSR register to 1.
b. Set the WRDSEL field of the MMSR register to 7.
c. Set the PRMADR field of the MMPAR register to 0.
d. Read the MAP B-BUS register (MMBUS).
e. Read the MAP B-BUS register and compare the results with the standard microcode table. If the results are not equal, the following message is output.

4: PROM/RAM DATA ERR; ADDR=XXXXXX EXPT=XXXXXX RECD=XXXXXX INDEX=37
P/R ADDR=m WSF=n

where m is the PROM/RAM address and n is the content of the WRDSEL field.

f. Do step e five more times.

g. Do step d two more times.

h. Do steps e, f, and g 255 more times.

i. Stop - phase complete.

Error Analysis:

Error #	Prognosis
1.	This error indicates the PRMADR cannot be written and read correctly. The PRMADR register is on the 195118-100 card. ADDR = the PICTURE SYSTEM address of PRMADR. EXPT = data the test is expecting from PRMADR. RECD = data the test received from PRMADR.
2.	This error indicates that the PRMADR (if INDEX=41) or the WRDSEL (if INDEX=37) did not increment correctly. The PRMADR register is on the 195118-100 card, and the WRDSEL register is on the 195117-100 card. ADDR = the PICTURE SYSTEM SCB address of the PRMADR. EXPT = data the test is expecting from PRMADR or WRDSEL. RECD = data the test received from PRMADR or WRDSEL.
3.	This error indicates that the DOIT register did not get written or read correctly. Of the 16 bits of expected data, the low eight bits correspond to

the DOIT bits of the 195115-101 card, the high eight bits correspond to the DOIT bits of the 195115-100 card.

ADDR = the PICTURE SYSTEM SCB address of the MAP B-BUS.
EXPT = data the test is expecting from DOIT register.
RECD = data the test received from DOIT register.
WSF = which set of eight bits are being tested.

WSF	DOIT bits
2	7-0
3	15-8
4	23-16
5	31-24
6	39-32
7	47-40

4. This error indicates the PROM store (read by testing the DOIT bits) is in error. Of the 16 bits of expected data, the low eight bits correspond to the DOIT bits of the 195115-101 card, the high eight bits correspond to the DOIT bits of the 195115-100 card.

ADDR = the PICTURE SYSTEM SCB address of MAP BBUS.
EXPT = data the test is expecting from DOIT register.
RECD = data the test received from the DOIT register.

WSF	DOIT bits
0	47-40
1	39-32
2	31-24
3	23-16
4	15-8
5	7-0

Name: (MAP INPUT CONTROLLER) FIFO & EXTEND REGISTER TEST

Functional Description:

This diagnostic tests the MAP input FIFO, input sign extension, and EXTEND input register. The diagnostic consists of three phases:

1. Input FIFO to B-BUS
   Loads the input FIFO with a fixed data sequence, and verifies that the data is input properly.
2. Input Sign Extension
   Loads the input FIFO with positive and negative values, checking that the input data was sign extended properly.
3. EXTEND Register Load
   Loads the input FIFO in extended mode (24 bits of input data), and verifies the EXTEND register.

Program Description:

Phase 1: Input FIFO test

a. Load input FIFO with 0, 177777, 125252, 052525.
b. Repeat steps c through f four times, using the expected data sequence (1) 0, (2) 177777, (3) 125252, (4) 052525.
c. Select DOIT to enable INPUT onto BBUS.
d. Test BBUS
e. If BBUS does not contain the expected data, the following message is output.

1: FIFO LOAD ERR; ADDR=XXXXX, EXPT=XXXXXX, RECD=XXXXXX, INDEX=37

f. Clock.
g. Load input FIFO with 177777, 0, 052525, 123456.

h. Repeat steps i through l four times, using the expected data sequence (1) 177777, (2) 0, (3) 052525, (4) 123456.

i. Select DOIT to enable INPUT onto BBUS.

j. Test BBUS

k. If BBUS does not contain the expected data, the following message is output.

1: FIFO LOAD ERR; ADDR=XXXXXX EXPT=XXXXXX RECD=XXXXXX INDEX=72

l. Clock.

m. End of phase.

Phase 2: Test the sign extension of the high order input bits and check for zero in the low order bits.

a. Set INPUT FIFO to 0.

b. Select DOIT to enable INPUT onto BBUS.

c. Test BBUS extended for 0. If an error occurs, the following message is output.

2: SIGN EXTEND ERR; ADDR=XXXXXX EXPT=XXXXXX RECD=XXXXXX INDEX=23

d. Clock.

e. Set INPUT FIFO to 100000.

f. Select DOIT to enable INPUT onto BBUS.

g. Test BBUS extended for 3. If an error occurs the following message is output.

2: SIGN EXTEND ERR; ADDR=XXXXXX EXPT=XXXXXX RECD=XXXXXX INDEX=30

h. End of phase.

Phase 3: Test loading and unloading of the extend register.

a. Repeat steps b through e five times, using the data sequence (1) 0, (2) 377, (3) 252, (4) 125, (5) 456.

b. Load INPUT FIFO with data pattern, and a "garbage" word.

c. Select DOIT to load EXTEND register from INPUT; clock.
d. Select DOIT to enable EXTEND onto BBUS.
e. Test BBUS extended for EXPECTED DATA. If an error occurs the following message is output.

3: EXTEND REG ERR; ADDR=XXXXXX EXPT=XXXXXX RECD=XXXXXX INDEX=33

f. End of phase.

Error Analysis:

Error #	Prognosis
1.	(index = 37,72) This error indicates the input FIFO cannot be read correctly from the MAP B-BUS. If only a few bits are in error, then the data path should be checked. If most of the bits are in error, the input controller should be checked. The input FIFO and controller are on the 195119-100 card. The data path to read the B-BUS is on the 195120-100 card. ADDR = the PICTURE SYSTEM SCB address of the MAP BBUS EXPT = data the test is expecting from the input FIFO. RECD = data the test received from the input FIFO.
2.	(index = 17,30) This error indicates the sign extension circuit on the 195120 card is not working properly.

ADDR = the PICTURE SYSTEM SCB address of the MAP BBUS

EXPT = data the test is expecting from the input FIFO.

RECD = data the test received from the input FIFO.

3. (index = 33)
This error indicates the EXTEND register on the 195120-100 card cannot be loaded or read correctly.

ADDR = the PICTURE SYSTEM SCB address of the MAP BBUS

EXPT = data the test is expecting from the input FIFO.

RECD = data the test received from the input FIFO.

Name: RSR DISPATCH TEST

Functional Description:

This test consists of one phase which loads data into the MAP Maintenance Repeat Status Register (MMRSR: 177755) and verifies the resulting contents of the PROM/RAM address register (MMPAR: 177756)

Program Description:

a. Repeat steps b through e using the input data sequence 0, 4, 10, 14, 20, 24, 30, 34, 40, 44, 50, 54, 60, 64, 70, 74 and the output (MMPAR) sequence 354, 345, 336, 332, 330,301,320,325,265, 255,250,220, 0, 0, 0, 0.
b. Load the input data into the 8 high-order bits of the MAP Input FIFO.
c. Select DOIT to: select input to BBUS, load RSR; Clock.
d. Select DOIT to: dispatch on RSR.
e Test MMPAR for the expected output data. If an error occurs, output the following message:

1: RSR DSPCH ERR; INDEX=32

f. End of test.

Error Analysis:

Error #	Prognosis
1	An error in this test suggests the Dispatch ROM on the 195118-100 card is failing. If QSD007 also fails, the error will be the RSR register on the 195120-100 card.

Name: RSR REGISTER TEST

Functional Description:

This test verifies loading and sequencing of the MAP Maintenance Repeat Status Register (MMRSR: 177755) in two phases:

1. Phase 1 loads the RSR (via the MAP input FIFO) and verifies the operation (via the MMRSR).
2. Phase 2 tests the RSR (or MAP finite state machine) sequencing.

Program Descripton:

Phase 1:
a. Repeat steps b through f using the input/output data sequence 0, 377, 252, 125, 456.
b. Load input data into the 8 high-order bits of the MAP input FIFO.
c. Select DOIT to: load RSR register from input; Clock.
d. Test RSR (MMRSR bits 15-8) for expected output data. If an error occurs, output the following message:

1: RSR LOAD ERR; INDEX=26

e. Select DOIT to: No operation; Clock.
f. The contents of the RSR should be unchanged. If an error occurs, output the message:

1: RSR LOAD ERR; INDEX=31

g. End Phase 1.

Phase 2:
a. Repeat steps b through e using the input data sequence 300, 311, 322, 333, 344, 355, 366, 377, and the output data sequence 311,310,333, 333, 344, 354,367, and 376.

b. Load input data into the 8 high-order bits of the MAP input FIFO.

c. Select DOIT to: load RSR register from input FIFO; Clock.

d. Select DOIT to: update RSR; Clock.

e. Test RSR (MMRSR bits 15-8) for the expected output data. If an error occurs, output the message:

2: FSM INCR ERR; INDEX=32

f. End of test.

Error Analysis:

Error #	Prognosis
1.	An error in this test suggests the RSR register on the 195120-100 card can not be written and/or read.
2.	An error in this test (assuming phase 1 passes), suggests the RSR UPDATE PROM on the 195120-100 card is in error.

Name: (MAP) SUBROUTINE CALL & RETURN TEST

Functional Description:

This program consists of one phase which verifies the PROM/RAM subroutine call and return facility.

Program Description:

a. Perform steps (b) through (f) with the data sequence 0, 377, 252, 125, 247.
b. Place next data pattern in NBADR field of DOIT; select DOIT to load subroutine register (*LDSUB=0); clock.
c. Select DOIT to enable subroutine return (*SELSUBRTN=0).
d. Test PROM/RAM address for the expected data pattern. If an error occurs, output the following message:

1: CALL/RETURN ERR; INDEX=31

e. Clock. Select DOIT to enable subroutine return.
f. Test PROM/RAM address for the expected data pattern. This ensures that the clock in step (e) did not load the subroutine return register. If an error occurs, output the following message:

1: CALL/RETURN ERR; INDEX=36

g. End of test

Error Analysis:

Error	Prognosis
1	This error occurs when the subroutine return address register (located in the lower right hand corner of sheet 5 of the 195118-100 card) can not be written or read correctly. If error index 36 occurs (but not error index 31) the clock enable of the register is stuck such that the register is always loading.

Name: MAP COUNTER AND TRANSPOSE TESTS

Functional Description:

This program tests the MAP counters and associated data paths, and the Data RAM address transposition capability (used in matrix transposition). The phases of the test are as follow:

1. DOIT/RAMADR data path
2. DOIT/AC0, AC1, AC2, AC3 data path
3. BBUS/RAMADR data path
4. Transposition from DOIT
5. Transposition from counter AC0

Program Description:

Phase 1: DOIT/RAMADR data path

a. Repeat steps (b) through (e) for the data sequence 0, 377, 252, 125, 056.

b. Load the DOIT subfield REG(7:0) with the next data pattern; select DOIT for REG(7:0)→RAMADR→BBUS (*MNTOUT=0);

c. Test the COUNT subfield of MMRSR (SCB: 177755; bits 15:8) for the current data pattern. If an error occurs, output the following message:

1: DOIT/RAMADR LD ERR; INDEX=30

d. Read and test MMBUS (SCB: 177757) for the current data pattern sign extended. If an error occurs, output the following message:

1: DOIT/RAMADR LD ERR; INDEX=33

e. Read and test the EXTBUS subfield of MMPAR (SCB: 177756; bits 15:8) bits 15:10 should be cleared, and bits 8:9 should be the sign-exten-

sion of the current data pattern. If an error occurs, output the following message:

1: DOIT/RAMADR LD ERR; INDEX=36

f. End of Phase

Phase 2: DOIT/AC0, AC1, AC2, AC3 data path

a. Repeat steps (b) through (n) for the data sequence 0, 377, 252, 125, 056.

b. Load DOIT subfield REG(7:0) with the next data pattern; select DOIT for REG→AC1, AC0; clock.

c. Load REG with the current data pattern; select DOIT for REG→AC1, AC0→RAMADR, increment AC0. Test COUNT (cf. Phase 1 step c.) for the current data pattern. If an error occurs, output the following message:

2: COUNTER ERR; INDEX=57

d. Clock. Select DOIT for AC0→RAMADR. Test COUNT for pattern +1. If an error occurs, output the message:

2: COUNTER ERR; INDEX=60

e. Load REG with data pattern. Select DOIT for REG→AC2, AC1→RAMADR, increment AC1, increment AC0. Test COUNT for the current data pattern. If an error occurs, output the following message:

2: COUNTER ERR; INDEX=65

f. Clock. Select DOIT for AC1→RAMADR. Test COUNT for pattern +1. If an error occurs,

output the following message:

2: COUNTER ERR; INDEX=74

g. Select DOIT for AC0→RAMADR. Test COUNT for pattern +2. If an error occurs, output the following message:

2: COUNTER ERR; INDEX=101

h. Load REG subfield of DOIT with the current data pattern. Select DOIT for REG→AC3, AC2→RAMADR, increment AC2, increment AC1. Test COUNT for the current data pattern. If an error occurs, output the following message:

2: COUNTER ERR; INDEX=110

i. Clock. Select DOIT for AC2→RAMADR. Test COUNT for pattern +1. If an error occurs, output the following message:

2: COUNTER ERR; INDEX=117

j. Select DOIT for AC1→RAMADR. Test COUNT for pattern +2. If error, then output the message:

2: COUNTER ERR; INDEX=123

k. Select DOIT for AC3→RAMADR, increment AC2, increment AC3. Test COUNT for pattern. If error, output the message:

2: COUNTER ERR; INDEX=132

1. Clock. Select DOIT for AC3→RAMADR. Test COUNT for pattern +1. If error, output the message:

2: COUNTER ERR; INDEX=141

m. Select DOIT for AC2→RAMADR, increment AC3. Test COUNT for pattern +2. If error occurs, output the following message:

2: COUNTER ERR; INDEX=146

n. Clock. Select DOIT for AC3→RAMADR. Test COUNT for pattern +2. If error, output the message:

2: COUNTER ERR; INDEX=153

o. End phase 2.

Phase 3: BBUS/RAMADR data path.

a. Repeat steps (b) through (c) for the data sequence 0, 377, 252, 125, 056.
b. Select DOIT for INPUT→BBUS→AC0. Load next data pattern into input FIFO. Clock.
c. Select DOIT for AC0→RAMADR. Test COUNT (cf. Phase 1 , step c) for the current data pattern. If an error occurs, output the message:

3: B-BUS/RAMADR ERR; INDEX=31

d. End Phase 3.

Phase 4: Check the control logic which causes the low-order four bits of RAMADR to be transposed (exchange bits 3 and 1, and exchange bits 2 and 0 - used for matrix transposition).

a. Select DOIT for REG=11, TRANSPOSE (*TRANSPOSE=0).

Test COUNT for 6. If error, output the message:

4: DOIT TRANSPOSE ERR; INDEX=26

b. Select DOIT for REG=11, TRANSPOSE, CHECK TRANSPOSE (*CHECKSTATUS=0). Test COUNT for 6. If error, output the message:

4: DOIT TRANSPOSE ERR; INDEX=27

c. Set input FIFO to 1000. Select DOIT for INPUT→RSR. Clock.
d. Select DOIT for REG=6. Test COUNT for 6. If error, output the message:

4: DOIT TRANSPOSE ERR; INDEX=36

e. Select DOIT for REG=6, TRANSPOSE. Test COUNT for 11. If error, output the message:

4: DOIT TRANSPOSE ERR; INDEX=42

f. Select DOIT for REG=6, TRANSPOSE, CHECK TRANSPOSE. Test COUNT for 6. If error, output the message:

4: DOIT TRANSPOSE ERR; INDEX=44

Phase 5: Check transposition of counters.

a. Select DOIT for REG=11, REG→AC0. Clock.
b. Select DOIT for TRANSPOSE, AC0→RAMADR. Test COUNT for 6. If error, then output the message:

5: AC0 TRANSPOSE ERR; INDEX=17

c. Select DOIT for REG=6, REG→AC0. Clock.

d. Select DOIT for TRANSPOSE, AC0→RAMADR. Test COUNT for 11. If error, output the message:

5: AC0 TRANSPOSE ERR; INDEX=26

e. End of test.

Error Analysis:

All phases of this test check the data paths on the 195114-100 card.

Error	Prognosis
1:	If an error occurs in Phase 1, the multiplexers on the bottom of sheet 4 are the cause. If INDEX=33, 36 in Phase 1 then the BBUS drivers on sheet 5 will be the cause of the error.
2:	INDEX=51; then AC0 counter cannot be written and/or read. INDEX=60; then AC0 counter cannot count. INDEX=65; then AC1 counter cannot be written and/or read. INDEX=74; then AC1 counter cannot count. INDEX=101; then AC0 counter cannot count. INDEX=110; then AC2 counter cannot be written and/or read. INDEX=117; then AC2 counter cannot count. INDEX=123; then AC1 counter cannot count. INDEX=132; then AC3 counter cannot be written and/or read. INDEX=141; then AC3 counter cannot count. INDEX=146; then AC2 counter cannot count. INDEX=153; then AC3 counter cannot count. All Phase 2 errors use the data path from MCREG to counters (sheet 3), then from counters to ADR (sheet 4 top) to RAMADR (sheet 4 bottom).

Error Analysis:

Error	Prognosis
3:	This phase checks the data path from the BBUS to AC0 counter (sheet 3), then to ADR and then RAMADR (sheet 4).
4:	All parts of this phase check the transpose logic and multiplexers on the lower half of sheet 4. See the program description of this phase for conditions of *TRANSPOSE and *CHECKSTATUS. The test checks the data path from MCREG through the transpose logic to RAMADR.
5:	This test is the same as Phase 4, except the data path is from MCREG to AC0, from AC0 to ADR, and from ADR through the transpose logic to RAMADR.

Name: MAP DATA STORAGE CARD TEST

Functional Description:

In 5 phases, this test checks data paths and registers on the Data Storage Card (195116-100). For phases 1-3, the 24-bit data sequence is 0, 77777777, 25252525, 52525252, 01234567. For all error messages which have two indices associated with them, the first index refers to B-BUS EXTEND data, and the second refers to B-BUS data.

Program Description:

Phase 1: Test loading of MB, MDA, and RA from CBUS.

a. Repeat steps (b) through (s) five times, using the 24-bit data sequence.
b. Set INPUT FIFO to data pattern (2 words).
c. Select DOIT to: LOAD EXTEND Register from input; 0→CBUS→MDA,RA. Clock.
d. Select DOIT to: EXTEND mode; INPUT→BBUS→ GBUS→CBUS→MB. Clock.
e. Select DOIT to: MB→BBUS. Test for expected data. If error, then output the message:

1: MB/MDA/RA LD ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX= 37 (or 42)

f. Select DOIT to MDA→BBUS. Test for 0. If error, then output:

1: MB/MDA/RA LD ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=51 (or 45)

g. Select DOIT to: RA→CBUS→BBUS. Test for 0. If error, then output:

1: MB/MDA/RA LD ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=55 (or 60)

h. Set INPUT FIFO to data pattern (2 words).
i. Select DOIT to: LOAD EXTEND Register from INPUT; 0→CBUS→MB, RA. Clock.
j. Select DOIT to: EXTEND mode; INPUT→BBUS→GBUS→CBUS→MDA. Clock.
k. Select DOIT to: MB→BBUS. Test for 0. If error, then output:

1: MB/MDA/RA LD ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=104 (or 101)

l. Select DOIT to: MDA→BBUS. Test for data pattern. If error, then output:

1: MB/MDA/RA LD ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=111 (or 114)

m. Select DOIT to: RA→CBUS→BBUS. Test for 0. If error:

1: MB/MDA/RA LD ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=123 (or 117)

n. Load INPUT FIFO with data pattern (2 words).
o. Select DOIT to: LOAD EXTEND Register from INPUT; 0→CBUS→MB,MDA. Clock.
p. Select DOIT to: EXTEND mode; INPUT→BBUS→GBUS→CBUS→RA. Clock.
q. Select DOIT to: MB→BBUS. Test for 0. If error:

1: MB/MDA/RA LD ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=144 (or 147)

r. Select DOIT to: MDA→BBUS. Test for 0. If error:

1: MB/MDA/RA LD ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=156 (or 152)

s. Select DOIT to: RA→CBUS→BBUS. Test for data pattern. If error:

1: MB/MDA/RA LD ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=163 (or 165)

t. End Phase 1.

Phase 2: Test data path from BBUS to DATA MEMORY (Memory location 0 only).

a. Repeat steps (b) through (i) five times, using the 24-bit data sequence given above.
b. Load data pattern into INPUT FIFO (2 words).
c. Select DOIT to: LOAD EXTEND Register from INPUT. Clock.
d. Select DOIT to: EXTEND mode; INPUT→BBUS→MEMORY. Clock.
e. Select DOIT to: MEMORY→ABUS→BBUS; MEMORY→ABUS→MB,MDA,RA.
f. Test BBUS for data pattern. If error, then output the message:

2: RAM/ABUS ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=37 (or 42)

g. Clock. Select DOIT to: MB→BBUS. Test BBUS for data pattern. If error:

2: RAM/ABUS ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=54 (or 57)

h. Select DOIT to: MDA→BBUS. Test BBUS. If error:

2: RAM/ABUS ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=70 (or 62)

i. Select DOIT to: RA→CBUS→BBUS. Test BBUS. If error:

2: RAM/ABUS ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=73 (or76)

j. End Phase 2.

Phase 3: Test data path from MB to DATA MEMORY (location 0 only).

a. Repeat steps (b) through (f) five times, using the data sequence above.
b. Set INPUT FIFO to data pattern (2 words).
c. Select DOIT to: LOAD EXTEND Register from INPUT. Clock.
d. Select DOIT to: EXTEND mode; INPUT→BBUS→GBUS→CBUS→MB. Clock.
e. Select DOIT to: MB→MEMORY. Clock.
f. Select DOIT to: MEMORY→ABUS→BBUS. Test BBUS for expected data. If error, then output the message:

3: MB/RAM LOAD ERR; INDEX=37 (or 43)

g. End of Phase 3.

Phase 4: RAM content; Data Pattern 1 (0707.., 2525.., 5252..).

a. Reset the Map.
b. FIFO←07070707 (2 words). Select DOIT to: LOAD EXTEND. Clock.
c. Select DOIT to: EXTEND MODE; INPUT→BBUS→GBUS→CBUS→MB. Clock. (Effect of steps b & c: MB←07070707).
d. In similar fashion, MDA←25252525 and RA←52525252. AC0←0.
e. Select DOIT to: MB→BBUS→MEMORY(AC0); increment AC0. Clock (load MEMORY location 0).
f. Repeat steps (g) through (i) 85 times (Load MEMORY locations 1-255).
g. Select DOIT to: MDA→BBUS→MEMORY(AC0); increment AC0. Clock.
h. Select DOIT to: RA→CBUS→BBUS→MEMORY(AC0); increment AC0. Clock.

i. Select DOIT to: MB→BBUS→MEMORY(ACO); increment AC0. Clock.

j. Repeat step (k) 85 times (test locations 0-254).

k. Repeat step (1) three times using the data sequence 07070707, 25252525, 52525252.

1. Select DOIT to: MEMORY(AC0)→ABUS→BBUS; increment AC0. Test for expected data and output the following message if an error occurs, and then clock:

4: RAM CONTENT ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=100 (or 103); RAMADR=nnn

m. In similar fashion, test location 255 for 07070707. If an error occurs, output the message:

4: RAM CONTENT ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=122 (or 125); RAMADR=377

n. End Phase 4.

Phase 5: RAM Content; Data Pattern 2 (70707070, 52525252, 25252525).

a. The steps of Phase 5 are identical to those of Phase 4, substituting Data Pattern 2 for Data Pattern 1. The error messages are also the same, except that the error identification number is 5 instead of 4:

5: RAM CONTENT ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=100 (or 103); RAMADR=nnn

5: RAM CONTENT ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=122 (or 125); RAMADR=377

b. End of test.

ERROR ANALYSIS:

This is intended to clarify the Error Analysis on QSD010. Standard error message:

Phase number: Error Type;ADDR=xxxxxxEXPT=xxxRECD=xxxINDEX=xxx

The Phase number and the Index number will tell thru use of the prognosis, if the problem is in a register, a loading problem or in the Randon Access Memory (RAM).

The Address and the 16 bit received word (in octal) will tell thru use of the charts below, which, bit on the 195116 card is bad and which slot the bad card is in.

6531 chips

To Use Charts:

A. Only use the first error message (standard error message) printed out.

B. Address 177756 (standard error message) refers to Chart "1".

C. Address 177757 (standard error message) refers to Chart "2".

D. Top line of charts is binary representation of 16 bit (octal) rec'd word (standard error message).

E. Numbered spaces below the 16 bit rec'd word refer to bits labeled (11-0) on 195116-600 logic drawings.

CHART 1

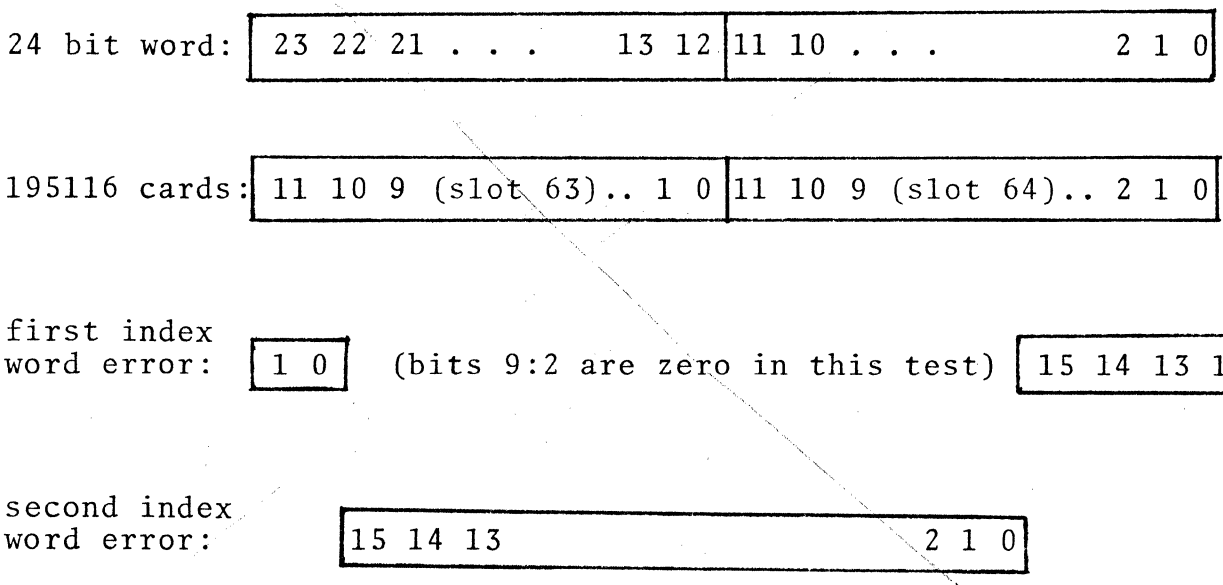
Rec'd word (binary)	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Slot 63 bits (11-10)							11	10	NOT							
Slot 64 bits (5-0)	5	4	3	2	1	0		USED								

CHART 2

Rec'd word (binary)	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Slot 63 bits (9-0)	9	8	7	6	5	4	3	2	1	0						
Slot 64 bits (11-6)											11	10	9	8	7	6

Error Analysis: delete

This test checks the Data Storage Cards (195116-100). Two 12-bit slice cards are used in the system. In order to map from the error message to the system card, the following diagram should be used.



Error #	Index #	Prognosis
1	37, 42 104, 101 144, 147	MB register on the 195116-100 card (Sheet 3) cannot be written/and or read from the BBUS correctly.
1	51, 45 111, 114 156, 152	MDA register on the 195116-100 card (Sheet 3) cannot be written/and or read from the BBUS correctly.
1	55, 60 123, 117 163, 165	RA register on the 195116-100 card (Sheet 3) cannot be written/and or read from the BBUS correctly.
2	37, 42	The 256 word RAM on the 195116-100 card (Sheet 5) cannot be written/and or read from the BBUS correctly.

Error #	Index #	Prognosis
2	54, 57	The MB register could not be loaded from the RAM correctly.
2	70, 62	The MDA register could not be loaded from the RAM correctly.
2	73, 76	The RA register could not be loaded from the RAM correctly.
3	37, 43	The RAM could not be loaded correctly from the MB register.
4	100, 103 122, 125	The RAM memory cannot store values correctly (i.e., the memory is in error). RAMADR is the memory location that failed.
5	100, 103 122, 125	The RAM memory cannot store values correctly (i.e., the memory is in error). RAMADR is the memory location that failed.

Name: (MAP) ALU TESTS

Functional Description:

This test consists of one phase which tests the MAP ALU (Arithmetic & Logic Unit). Random numbers are loaded into MDA and RA. Then for each set of random numbers, the FUNC0, FUNC1, FUNC2, FUNC3, LOGIC and CARRYIN control lines are stepped through all 64 combinations.

Program Description:

Phase 1:

a. Reset MAP
b. Perform steps (c) through (h) 99 times.
c. Put four random numbers in the FIFO.
d. Select DOIT to: LOAD EXTEND register from input. Clock.
e. Select DOIT to: EXTEND mode; INPUT→BBUS→GBUS→CBUS→MDA. Clock. (Effect: MDA←random number composed of first two words in FIFO).
f. Select DOIT to: LOAD EXTEND register from INPUT. Clock.
g. Select DOIT to: EXTEND mode; INPUT→BBUS GBUS→CBUS→RA. Clock. (Effect: RA←next two words in FIFO).
h. Perform steps (i) through (l) 64 times using the data sequence (f = 0 through 77) for the 6 ALU control bits (M, *CN, S0-S3).
i. Perform ALU simulation to predict result.
j. Select DOIT to: M, *CN, S0-S3 from current data pattern (f); MDA→BBUS→GBUS; result of ALU (f,GBUS,RA)→MB. Clock.
k. Select DOIT to: MB→BBUS. Test BBUS EXTEND for predicted result. If error, then output the message:

1: ALU RESULT ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=55
MDA=nnnnnnnn RA=nnnnnnnn RESULT=nnnnnnnn (M,*CN,S0-S3)=nn

Where RESULT is the 24-bit simulator result. Note the EXPT and RECD are in BBUS EXTEND format such that bits 1-0 correspond to bits 23-22 and bits 15-10 correspond to bits 5-0 of the 24 bit value.

l. Test BBUS for expected result. If error, then output the message:

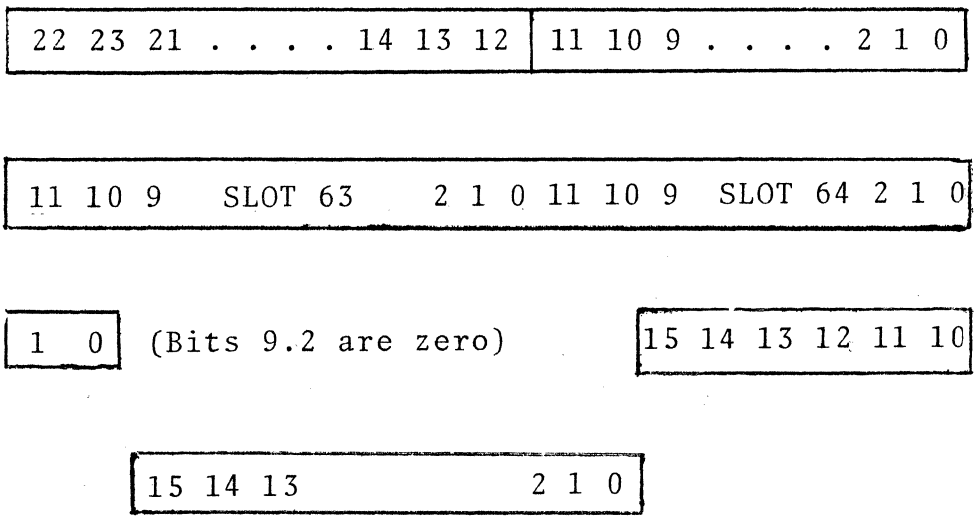
1: ALU RESULT ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=61
MDA=nnnnnnnn RA=nnnnnnnn RESULT=nnnnnnnn (M,*CN,S0-S3)=nnnn

Note that in this case, EXPT and RECD correspond to bits 21-6 of the 24 bit MPA value.

m. End of test.

Error Analysis

Error #	Prognosis
1	This test checks all possible functions of the ALU (74S181, 74S182) on the 195116-100 card (Sheet 2). The MDA register drives the 'A' side of the ALU, and the RA register drives the 'B' side. The RESULT is the value that should appear at the 'F' output of the ALU (or the 'C' BUS). The (M, *CN, S0-S3) field contains the data used to control the ALU. In order to map from the error message to the system card, the following diagram should be used.



Identification: QSD012
Date: 3/2/77

Name: MAP NORMALIZE SENSE TESTS

Functional Description:

In four phases, this test checks the NORMALIZE SENSE, SENSE REGISTER, and SENSE COMPARATOR (195114-100).

Program Description:

Phase 1: Positive number test

a. Repeat steps (b) through (f) 24 times, using the double-word input sequence 0(0,0), 1(2000,0), 2,(4000,0), ...40,(100000,0), 100(0,1), 200(0,2), ...10000000(0,100000), 20000000(1,0). In the foregoing context, values not in parenthesis are 24-bit MAP data values, and pairs of values in parenthesis are the corresponding EXTEND followed by normal INPUT values. (Example: to load the 24-bit value 200, load 0 into the EXTEND register, then load 2 into INPUT register). Also, for the 24 (decimal) iterations, use the expected octal data sequence 27 to 0.

b. Load the next data pattern (2 words) into the FIFO.

c. Select DOIT to: LOAD EXTEND Register from INPUT. Clock.

d. Select DOIT to: EXTEND mode; INPUT→BBUS; Load SENSE REGISTER. Clock.

e. Select DOIT to: SENSE REGISTER→MASHFC→BBUS. Test BBUS for expected data. If error:

1: NORM SENSE ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=43 INPUT=nnnnnnnn

f. Select DOIT to: "2" →MASHFC→BBUS. Test BBUS for 2. If error:

1: NORM SENSE ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=50 INPUT=nnnnnnnn

g. End of phase 1.

Phase 2: Negative Number Test

a. Repeat steps (b) through (e) 24 times with the 24-bit input data sequence 77777777, 77777776, 77777775, 77777773, 77777767, ... 73777777, 67777777, 57777777. The expected output data sequence is 27-0. Steps (b) through (e) are identical to those of phase 1, except for the error message of step (e).

b. Load the next data pattern (2 words) into the FIFO.

c. Select DOIT to: LOAD EXTEND Register from INPUT. Clock.

d. Select DOIT to: EXTEND mode; INPUT→BBUS; Load SENSE REGISTER. Clock.

e. Select DOIT to: SENSE REGISTER→MASHFC→BBUS. Test BBUS for expected data. If error:

2: NORM SENSE ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=43 INPUT=nnnnnnnn

f. End of phase 2.

Phase 3: Sense Comparator Test A.

a. Load INPUT FIFO with 0 (2 words).

b. Select DOIT to: LOAD EXTEND register from INPUT. Clock.

c. Select DOIT to: EXTEND mode; INPUT→BBUS; LOAD SENSE REGISTER. Clock.

d. Repeat steps (e) through (h) with 24-bit input sequence 1, 2, 4,...10000000, 20000000, and expected output sequence 26 to 0.

e. Load INPUT FIFO with next data pattern (2 words).

f. Select DOIT to: LOAD EXTEND register from INPUT. Clock.

g. Select DOIT to: EXTEND mode; INPUT→BBUS; LOAD MINIMUM SENSE. Clock. (The sense register should load).

h. Select DOIT to: SENSE REGISTER→MASHFC→BBUS. Test BBUS for the next expected data pattern. If error:

3: NORM SENSE ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=54 INPUT=nnnnnnnn

i. End Phase 3.

Phase 4: Sense Comparator Test B.

a. Perform steps (b) through (h) 23 times. Each step of the input sequence consists of two 24-bit values (4 words total): $(2^{22},2^{21})$, $(2^{21}, 2^{20})$.... (1,0). Each 24-bit value must be loaded in the fashion of the preceding phases; e.g. to load 2^{20}, load 0 into the FIFO followed by 40000 octal. The expected output sequence is octal 0 to 26. As in the other phases, the output data is in 16-bit format, with the 6 least significant and two most significant bits truncated.

b. Load INPUT FIFO with the first 24-bit value of the next input pair.

c. Select DOIT to: LOAD EXTEND register from INPUT. Clock.

d. Select DOIT to: EXTEND mode; INPUT→BBUS; LOAD SENSE REGISTER. Clock.

e. Load INPUT FIFO with the second 24-bit value of the input pair.

f. Select DOIT to: LOAD EXTEND register from INPUT. Clock.

g. Select DOIT to: EXTEND mode; INPUT→BBUS; LOAD MINIMUM SENSE. Clock. (The sense register should not load).

h. Select DOIT to: SENSE REGISTER→MASHFC→BBUS. Test BBUS for the expected output. If error:

4: NORM SENSE ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=50
INPUT A=nnnnnnnn INPUT B=nnnnnnnn

i. End of test.

Error Analysis:

All parts of this test check normalize sense circuit on the 195114-100 card (Sheets 1, 2, 5).

Error#	Index#	Prognosis
1	43	The normalize sense (NS4-0) on Sheet 5, or the sense registers (Sheet 1-25S09's), or the mux selector (Sheet 1-74S157) may be in error.
1	50	This path only checks side B of the 74S157 mux (Sheet 1). MASHFC (4-0) is passed to the BBUS on Sheet 5 for diagnostic checking.
2	43	This test is the same as Phase 1 test except negative numbers are sensed. The XOR (74S86) on Sheet 2 senses the sign bit such that a negative number is always at BB22-0.
3	54	This test is similar to Phase 1, except initially the sense register is loaded with 27, and the successively smaller values (sense) are loaded. If the comparator (74S85 - Sheet 1) is working correctly the *MCNSRMINLD will load the successively smaller values.
4	50	This test is opposite of Phase 3, i.e., the successive inputs are larger, such that the sense register should not load when *MCNSRMINLD is asserted. This test is also checking the comparator (74S85 - Sheet 1).

Name: MAP MRA AND NORMALIZER TESTS

Functional Description:

Phase 1 of this test checks the MAP MRA register and associated data paths. Phase 2 checks the MAP normalizer.

Program Description:

Phase 1: MRA Data Path Test

a. Perform steps (b) through (g) 5 times with the 24-bit input/output data sequence 0, 37777777, 25252525, 52525252, 01234567.
b. Load the next data pattern into the INPUT FIFO (2 words).
c. Select DOIT to: LOAD EXTEND register INPUT. Clock.
d. Select DOIT to: EXTEND mode; INPUT→BBUS→MRA. Clock.
e. Select DOIT to: MRA→BBUS. Test BBUS EXTEND for the input data pattern. If an error occurs, then output the message:

1: MRA DATA ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=45

f. Test BBUS for the input data pattern. If error occurs, then output the message:

1: MRA DATA ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=51

g. End of phase 1.

Phase 2: Normalizer Test

a. Perform steps (b) through (e) 5 times with the input data sequence of Phase 1. Steps (f) through (k) are performed 5x24 times.
b. Load the next input pattern into the INPUT FIFO. (2 words).

c. Select DOIT to: LOAD EXTEND register from INPUT. Clock.

d. Select DOIT to: ENTEND mode; INPUT→BBUS→MRA. Clock.

e. Perform steps (f) through (k) 24 times with the 24-bit Sense Register data sequence 0, 1, $2...2^{22}$.

f. Perform Normalizer simulation (inputs are current MRA contents and next Sense Register data) to predict Normalizer output.

g. Load FIFO with the next Sense Register data pattern (2 words).

h. Select DOIT to: LOAD EXTEND register from INPUT. Clock.

i. Select DOIT to: EXTEND mode; INPUT→BBUS; LOAD SENSE REGISTER. Clock.

j. Select DOIT to: SENSE REGISTER→MASHFC; MRA→NORMALIZE→BBUS. Test BBUS EXTEND for predicted data. If error, then output the message:

2: NORMALIZER ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=66
INPUT=nnnnnnnn SENSE=nnnnnnnn NORM DATA S/B nnnnnnnn

k. Test BBUS for predicted data. If error:

2: NORMALIZER ERR; ADDR=XXXXXX EXPT=XXXXXX RECD=XXX INDEX=73
INPUT=nnnnnnnn SENSE=nnnnnnnn NORM DATA S/B nnnnnnnn

l. End of test.

Error Analysis:

On the 195114-100 card this test checks the MRA register (Sheet 1), the normalizer (Sheets 2, 3, 4) and the read back path of MRA and the normalizer to the BBUS (Sheet 5).

Error#	Index#	Prognosis
1	45	The high 2 bits and/or low 6 bits of the MRA register cannot be written and/or read correctly.
1	51	The remaining 16 bits (bits 21-6) of the MRA register cannot be written and/or read correctly.
2	66	The high 2 bits and/or low 6 bits of the normalized number cannot be read correctly. Note that the high 2 bits should be identical (NMR 23). See Sheet 5.
2	73	The remaining 16 bits (BBUS bits 21-6, or NMR bits 23-8) cannot be read correctly. For all ERROR#=2, the INPUT field is the number loaded into the MRA register, and SENSE is the number of bits to shift the INPUT field (MASHFC(4-0) on Sheets 2, 3, 4). NORMDATA S/B is the 24 bit expected result.

NAME: MAP RECIPROCAL ROM TEST

Functional Description:

In two phases, this test checks the contents of the MAP Reciprocal ROM. In the error messages which follow, the first index refers to a BBUS EXTEND error, and the second index refers to BBUS.

Program Description:

Phase 1: Positive MRA Rom Check

a. Select DOIT to: CLEAR EXTEND. Clock.

b. Repeat steps (c) through (f) 256 times using the 16-bit input data sequence 40000, 40100, 40200, 40300...77400, 77500, 77600, 77700. Expected output data is obtained from a software ROM table.

c. Load INPUT FIFO with next input data (1 word).

d. Select DOIT to: INPUT→BBUS→MRA. Clock.

e. Select DOIT to: R1ROM→BBUS. Test BBUS EXTEND and BBUS for expected data. If error:

1: RECIP ROM ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=37 (or 33) INPUT=nnnnn

f. Select DOIT to: R2ROM→BBUS. Test BBUS EXTEND and BBUS for expected data. If error:

1: RECIP ROM ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=44 (or 50) INPUT=nnnnn

g. End of Phase 1.

Phase 2: Negative MRA ROM Check

a. Set DOIT to: CLEAR EXTEND. Clock.

b. Repeat steps (c) through (f) 256 times using the 16-bit input data sequence 140000, 140100, 140200, 140300 ... 177400, 177500, 177600, 177700. Expected data is obtained from a software ROM table. Steps (c) through (f) are identical to those of Phase 1 except for error identification numbers.

c. Load INPUT FIFO with next input data (1 word).

d. Select DOIT to: INPUT→BBUS→MRA. Clock.

e. Select DOIT to: R1ROM→BBUS. Test BBUS EXTEND and BBUS for expected data. If error:

2: RECIP ROM ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=37 (or 33) INPUT=nnnnn

f. Select DOIT to: R2ROM→BBUS. Test BBUS EXTEND and BBUS for expected data. If error:

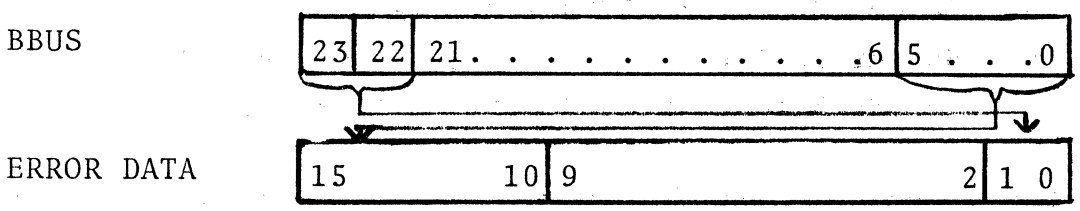
2: RECIP ROM ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=44 (or 50)

g. End of Test.

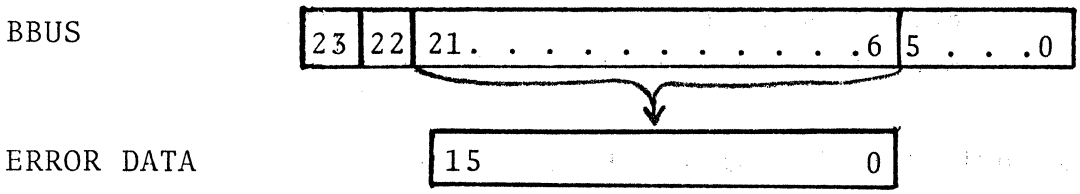
Error Analysis:

This test checks the reciprocal ROMS of the 195113-100 card, (Sheets 6, 7).
The following table should be used in the error prognosis:

INDEX=37,44



INDEX=33,50



Error #	Index #	Prognosis
1	37,33	BBUS bits 23, 22, 13-0 should always be zero. They are driven by *MT1 on Sheet 7. BBUS bits 21-14 are the contents of the R1ROM with NMR (23-14) being the address. INPUT in the error message is the expected address.
1	44, 50	BBUS bits 23, 6-0 should always be zero. They are driven by *MT2 on Sheet 7. BBUS bits 22-7 are the contents of the R2 ROM.
2	all	Same as Phase 1 but negative numbers are used for INPUT instead of positive numbers.

Name: MAP MULTIPLIER TEST

Functional Description:

This test consists of two phases which test the MAP Multiplier (195111-100, 195112-100). Phase 1 uses a predetermined input data sequence, and phase 2 uses random input data.

Program Description:

Phase 1: Fixed Data Test

a. Repeat steps (b) through (h) 16 times using the 24 bit input data sequence:

00000000	00000000
00000000	77777777
00000000	07777777
00000000	70000000
77777777	00000000
77777777	77777777
77777777	07777777
77777777	70000000
07777777	00000000
07777777	77777777
07777777	07777777
07777777	70000000
40000000	00000000
40000000	77777777
40000000	07777777
40000000	70000000

(This data sequence reads from left to right. However, as explained below, values in the left column are loaded into MDA, and values in the right column are loaded into MRA).

b. Load the next two 24 bit values (four words) into the FIFO.

c. Select DOIT to: LOAD EXTEND register from input. Clock.

d. Select DOIT to: EXTEND mode; INPUT→BBUS→GBUS→CBUS→MDA. Clock. (The effect of this and the preceding step is to load the first 24 bit value into MDA).

e. Similarly, load the next 24-bit value (two words of FIFO) into MRA.

f. Select DOIT to: MMP→GBUS→CBUS→BBUS. (Do not Clock).

g. Perform MAP Multiplier simulation to predict the contents of BBUS, and test BBUS. If error, then output the message:

1: MULTIPLY ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=52
MDA=nnnnnnnn MRA=nnnnnnnn PROD S/B nnnnnnnn

h. Perform simulation to predict BBUS EXTEND. Test BBUS EXTEND. If error:

1: MULTIPLY ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=57

i. End phase 1.

Phase 2: Random Data Test

a. Perform steps (b) through (h) 16 times.

b. Load four words of random data into the FIFO (two 24-bit values). The second 24 bit value, to be loaded into MRA, is in the range 0 to 07777777 or 70000000 to 77777777.

c. Select DOIT to: LOAD EXTEND register from input. Clock.

d. Select DOIT to: EXTEND mode; INPUT→BBUS→GBUS→CBUS→MDA. Clock. (The effect of this and the preceding step is to load the first 24-bit value into MDA).

e. Similarly, load the next 24-bit value (two words of FIFO) into MRA.

f. Select DOIT to: MMP→GBUS→CBUS→BBUS (Do not clock).

g. Perform MAP Multiplier simulation to predict the contents of BBUS, and test BBUS. If error, then output the message:

2: MULTIPLY ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=47
MDA=nnnnnnnn MRA=nnnnnnnn PROD S/B nnnnnnnn

h. Perform simulation to predict BBUS EXTEND. Test BBUS EXTEND. If error:

2: MULTIPLY ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=54
MDA=nnnnnnnn MRA=nnnnnnnn PROD S/B nnnnnnnn

i. End of test.

Error Analysis:

This test checks the multiplier contained on the 195111-100 and 195112-100 cards. The multiplier multiplies 16 bits from MRA (via the NMR normalizer) and 20 bits from MDA. The result is sent back to the 195116-100 cards on the MMP bus (24 bits).

Error #	Prognosis
1,2	First check the 195116-100 card (MMP bus). If data is correct at this point, the multiplexer on the 195116-100 (Sheet 4) that transfers MMP to the GBUS is in error. However, if data is incorrect at MMP, then the tedious task of checking the 2x4 multipliers must be done by hand. For the circuit, the AMD 25S05 reference book should be used.

Name: MAP DISPATCH CODE TEST

Functional Description:

The 20_8 phases of this test check the 20_8 MAP controller dispatch codes (octal 0-17). The cards involved are 195117-100 and 195118-100.

Program Description:

Phase 1: Dispatch Code 0.

a. Select DOIT to: NBADR=0; *DCODE=17.
b. Test PROM/RAM address for 0. If error, then output the message:

1: DISPATCH ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=15

Phase 2: Dispatch Code 1.

a. Select DOIT to: NBADR=0; *DCODE=16.
b. Test PROM/RAM address for 1. If error, then output the message:

2: DISPATCH ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=15

Phase 3: Dispatch Code 2.

a. Select DOIT to: AC2←REG; REG=0. Clock. (This loads 0 into AC2).
b. Select DOIT to: NBADR=0; *DCODE=15.
c. Test PROM/RAM address for 0. If error:

3: DISPATCH ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=17

d. Select DOIT to: AC2←REG; REG=377. Clock. (This loads 377 into AC2).
e. Select DOIT to: NBADR=0; *DCODE=15.
f. Test PROM/RAM address for 1. If error, then output the message:

3: DISPATCH ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=26

Phase 4: Dispatch Code 3.

a. Select DOIT to: AC3←REG; REG=0. Clock. (This loads 0 into AC3).
b. Select DOIT to: NBADR=0; *DCODE=13.
c. Test PROM/RAM address for 0. If error:

4: DISPATCH ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=17

d. Select DOIT to: AC3←REG; REG=377. Clock. (This loads 377 into AC3).
e. Select DOIT to: NBADR=0; *DCODE=14.
f. Test PROM/RAM address for 1. If error:

4: DISPATCH ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=26

Phase 5: Dispatch Code 4.

a. Reset MAP.
b. Select DOIT to: NBADR=0; *DCODE=13.
c. Test PROM/RAM address for 0. If error:

5: DISPATCH ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=15

d. Select DOIT to: Set output flag. Clock.
e. Select DOIT to: NBADR=0; *DCODE=13.
f. Test PROM/RAM address for 1. If error:

5: DISPATCH ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=23

Phase 6: Dispatch Code 5.

a. Load INPUT FIFO with 0.
b. Select DOIT to: MRA,B←INPUT FIFO. Clock.
c. Select DOIT to: NBADR=0; *DCODE=12.
d. Test PROM/RAM address for 0. If error:

6: DISPATCH ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=21

e. Load INPUT FIFO with -1.
f. Select DOIT to: MRA,B←INPUT FIFO. Clock.
g. Select DOIT to: NBADR=0; *DCODE=12.
h. Test PROM/RAM address for 1. If error:

6: DISPATCH ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=32

Phase 7: Dispatch Code 6.

a. Set MODE=0.
b. Select DOIT to: NBADR=0; *DCODE=11.
c. Test PROM/RAM address for 0. If error:

7: DISPATCH ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=15

d. Set MODE=3.
e. Select DOIT to: NBADR=0; *DCODE=11.
f. Test PROM/RAM address for 1. If error:

7: DISPATCH ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=23

Phase 10_8: Dispatch Code 7.

a. Set MODE=0.
b. Select DOIT to: NBADR=0; *DCODE=10.
c. Test PROM/RAM address for 0. If error:

10: DISPATCH ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=15

d. Set MODE=2.
e. Select DOIT to: NBADR=0; *DCODE=10.
f. Test PROM/RAM address for 1. If error:

10: DISPATCH ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=23

Phase 11_8: Dispatch Code 10.

a. Load INPUT FIFO with 0.
b. Select DOIT to: MDA,C,G,B←INPUT FIFO. Clock.
c. Select DOIT to: NBADR=0; *DCODE=7.
d. Test PROM/RAM address for 0. If error, then output the message:

11: DISPATCH ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=21

e. Load INPUT FIFO with -1.
f. Select DOIT to: MDA,C,G,B←INPUT FIFO. Clock.
g. Select DOIT to: NBADR=0; *DCODE=7.
h. Test PROM/RAM address for 377. If error, then output the message:

11: DISPATCH ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=32

Phase 12_8: Dispatch Code 11.

a. Load INPUT FIFO with 0.
b. Select DOIT to: MDA,C,G,B←INPUT FIFO; LOAD FLAGS. Clock.
c. Select DOIT to: NBADR=0; *DCODE=6.
d. Test PROM/RAM address for 1. If error, then output the message:

12: DISPATCH ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=21

e. Load INPUT FIFO with 0,-1.
f. Select DOIT to: MDA,C,G,B←INPUT FIFO. Clock
g. Select DOIT to: C,G,B,←INPUT FIFO; LOAD FLAGS. Clock.
h. Select DOIT to: NBADR=0; *DCODE=6.
i. Test PROM/RAM address for 0. If error:

12: DISPATCH ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=37

j. Load INPUT FIFO with -1.
k. Select DOIT to: MDA,C,G,B←INPUT FIFO; LOAD FLAGS. Clock.
l. Select DOIT to: NBADR=0; *DCODE=6.
m. Test PROM/RAM address for 377. If error:

12: DISPATCH ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=50

Phase 13_8: Dispatch Code 12.

a. Load INPUT FIFO with 0.
b. Select DOIT to: RSR,B←INPUT FIFO. Clock.
c. Select DOIT to: NBADR=0; *DCODE=5.
d. Test PROM/RAM address for 0. If error:

13: DISPATCH ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=21

e. Load INPUT FIFO with -1.
f. Select DOIT to: RSR,B←INPUT FIFO. Clock.
g. Select DOIT to: NBADR=0; *DCODE=5.
h. Test PROM/RAM address for 1. If error:

13: DISPATCH ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=32

Phase 14_8: Dispatch Code 13.

a. Load INPUT FIFO with 0,0.
b. Select DOIT to: CLEAR EXTEND; SENSE REGISTER, AC2,B←INPUT FIFO; MDA,C,G,B←INPUT FIFO. Clock.
c. Select DOIT to: B←INPUT FIFO; LOAD FLAGS. Clock.
d. Select DOIT to: NBADR=0; *DCODE=4.
e. Test PROM/RAM address for 0. If error:

14: DISPATCH ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=26

f. Load INPUT FIFO with -1.
g. Select DOIT to: B←INPUT FIFO; LOAD FLAGS. Clock.

h. Select DOIT to: NBADR=0; *DCODE=4.
i. Test PROM/RAM address for 1. If error:

14: DISPATCH ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=37

j. Load INPUT FIFO with 0, -1.
k. Select DOIT to: B←INPUT FIFO; LOAD FLAGS. Clock.
l. Select DOIT to: AC2,B←INPUT FIFO; MDA,C,G,B,←INPUT FIFO. Clock.
m. Select DOIT to: NBADR=0; *DCODE=4.
n. Test PROM/RAM address for 1. If error:

14: DISPATCH ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=54

o. Load INPUT FIFO with -1.
p. Select DOIT to: SELECT EXTEND; MDA,C,G,B←INPUT FIFO. Clock.
q. Select DOIT to: NBADR=0; *DCODE=4.
r. Test PROM/RAM address for 1. If error:

14: DISPATCH ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=65

Phase 15_8: Dispatch Code 14.

a. Load INPUT FIFO with 0, -1.
b. Select DOIT to: MDA,C,G,B←INPUT FIFO. Clock.
c. Select DOIT to: NBADR=0; *DCODE=3.
d. Test PROM/RAM address for 0. If error:

15: DISPATCH ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=23

e. Select DOIT to: MDA,C,G,B,←INPUT FIFO. Clock.
f. Select DOIT to: NBADR=0; *DCODE=3.
g. Test PROM/RAM address for 1. If error:

15: DISPATCH ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=32

Phase 16_8: Dispatch Code 15.

a. Reset MAP.
b. Select DOIT to: NBADR=0; *DCODE=2.
c. Test PROM/RAM address for 0. If error:

16: DISPATCH ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=15

d. Select DOIT to: JV. Clock.
e. Select DOIT to: NBADR=0; *DCODE=2.
f. Test PROM/RAM address for 1. If error.

16: DISPATCH ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=23

Phase 17_8: Dispatch Code 16.

a. Reset MAP.
b. Select DOIT to: NBADR=0; *DCODE=1.
c. Test PROM/RAM address for 0. If error:

17: DISPATCH ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=15

d. Load INPUT FIFO with -1.
e. Select DOIT to: ←INPUT FIFO; LOAD FLAGS. Clock.
f. Select DOIT to: LOAD FLAGS. Clock.
g. Select DOIT to: NBADR=0; *DCODE=1.
h. Test PROM/RAM address for 1. If error:

17: DISPATCH ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=30

Phase 20_8: Dispatch Code 17.

a. Reset MAP.
b. Load INPUT FIFO with 0,0,0.
c. Select DOIT to: NBADR=0; *DCODE=0.
d. Test PROM/RAM address for 0. If error:

20: DISPATCH ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=23

e. Load INPUT FIFO with 0.

f. Test PROM/RAM address for 1. If error:

20: DISPATCH ERR; ADDR=XXXXXX EXPT=XXX RECD=XXX INDEX=30

Error Analysis:

Error #	Prognosis
All errors	All Phases check the 74S251 selectors on sheet 3 of the 195117-100 card. The address for the selector is driven from field '*DCODE' or (*MCD, *MCC, *MCB, *MCA) on the logic drawing. For each of the phases, the value of *DCODE can be found in the previous Program Description. The output'XORADR' is XOR on sheet 5 of the 195118-100 card. This line then goes through 74S253 multipliers on sheet 4 to bus MDAD. MDAD is then driven back to PSDAT for software verification.

Name: MAP REGISTER STORAGE TEST (195116 card)

Functional Description:

This test verifies that the MAP registers are capable of repeated loading and storing without error. There is only one phase, which loads all 256 MAP registers with random numbers, swaps the top 128 with the bottom 128, several times, and then stores and compares to verify that the data is the same as was loaded.

If the phase is run repeatedly by the "DO" number of passes command, a different set of random numbers is sent to the MAP for each pass. If an error occurs, the program will loop using the set of random numbers which caused the failure.

Printed errors output the MAP register address, expected and received data.

Program Description:

There is only one phase to this test. It operates in the following manner:

a. Fill a 512 work buffer with a set of pseudo random numbers. Every time the phase is called, a new set is produced. Only restarting the program will reproduce the sequence of pseudo random numbers.
b. Send a "LOAD all 256 registers in full 24-bit format" RSR command to the MAP Passive Input Port.
c. Send the 256 random number pairs to the MAP PIP.
d. Output 50 "XFER Register" commands to the MAP PIP, to swap locations 0-127 with locations 128-255.
e. Send a "STORE all 256 registers in full 24-bit format" RSR command to the MAP PIP.
f. Fetch the 256 number pairs from the MAP PIP using the DMA.

Compare the table sent with the table retrieved, and flag the errors (see below).

Error Analysis:

Each register LOADS and STORES 24 bits of information, which is taken from and put on the PSBUS in 2 16-bit word sets. In the first word, only bits <1:0> and <15:10> are valid, and in the second word, all the bits <15:0> are valid, (drawing below).

If an error is encountered in the first word, only that word is typed out to the user. If an error is encountered in the second word, the entire number pair is typed out. The error message thus may take the form:

```
1:  ADDRESS     EXPECT     RECEIVE
       1        132003     112003     (:Error in first word only)
       4        14002      14002
                20743      20742      (:Error in second word only)
       15       144001     144000     (:Error in both first & second
                67521      66521       words)
```

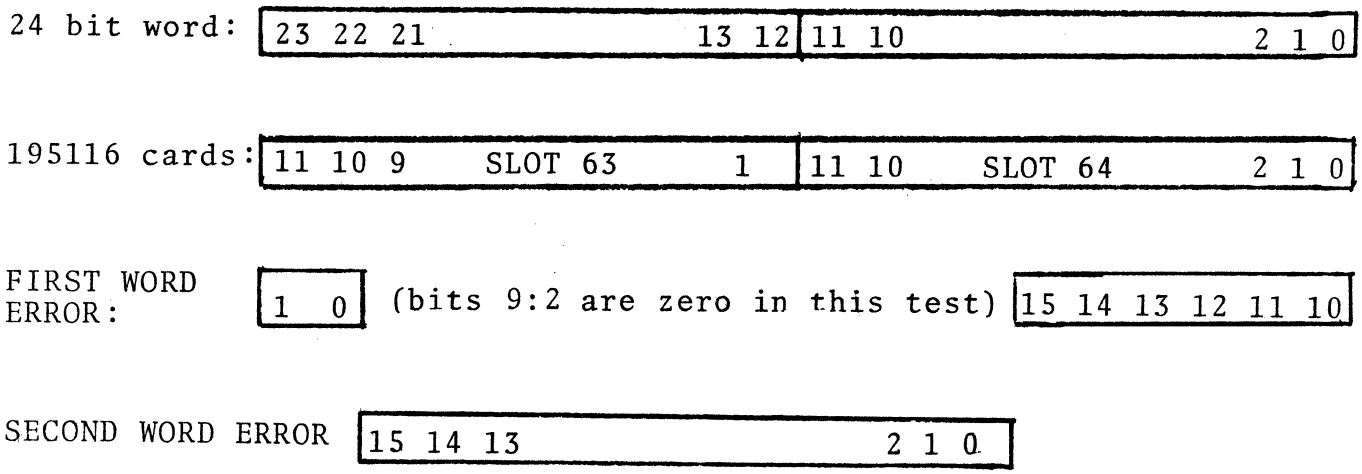
When looping on error, the program will send the same set of random numbers which caused the original error, but will only check the particular word (not word pair) in which an error was detected.

Prognosis:

This test is used for dynamically checking the 195116-100 cards. Particularly, it is used for verifying the 256 word RAM and the ALU's on that card. If only one bit (in one location) is in error, one would expect the RAM to be in

error. Otherwise the ALU or one of the multiplexors on the card should be suspected.

In order to map from the error message to the system card, the following diagram should be used.



Name: MAP OUTPUT FORMATTER TEST

Functional Description:

This test is used to verify that the MAP Output Formatter is capable of packing data correctly into the 12-bit and 16-bit formats used by the line generator and the driving program respectively. The seven possible uses of the commands 'Pass Formatted', 'Pass Conditional', and 'Pass' provide the seven phases as follows:

1. Pass Formatted, normal output
   Packs the data to be sent to the Line Generator with 12-bits of X and Y, and 6 bits of Z.
2. Pass Formatted, 16-bit precision.
   Keeps 16-bits of precision in X, Y and Z with the appropriate header word.
3. Pass, normal output.
   Keeps 16-bit of precision in X and Y. No header.
4. Pass, normal output
   Keeps 16-bits of precision in X, Y, and Z with the appropriate header word.
5. Pass Conditional, normal, HIT=1
   Lets the MAP assume a 'HIT' (as in 'hit testing' with the tablet) was made, and keeps 16-bits precision in X and Y with no header.
6. Pass Conditional, 16-bit precision, HIT=1
   Lets the MAP assume a 'HIT' was made, and keeps 16-bits precision in X, Y, and Z with the appropriate header word.
7. Pass Conditional, HIT=0
   Lets the MAP assume that the vector is outside the hit window, so no data should be returned.

In all phases, random data is sent to the MAP, and the output from the Output Formatter is compared against a reference word built from the data. Where the Output Formatter would produce different output for a 'MOVE' than for a 'DRAW', MOVE and DRAW commands are alternated.

NOTE: If the Real-Time Clock is broken, the test will not finish

Phase 1: Verify the command Pass Formatted, normal output. For this command, the Output Formatter expects 3 words of data input (being X, Y and Z), and will output 2 words of data (X, Y and Z packed).

a. Fetch 768 (=256*3) new random numbers.

b. Set the MMODE bits in the MSR to 0 (normal output).

c. Send to the MAP PIP the RSR command 124000, which is for 3-D data, FSM1=0 (MOVE, DRAW, MOVE, DRAW,....), FSM2=5 (Pass Formatted).

d. Set up the DMA to read back 512 (=256*2) words of data from the MAP POP whenever it gets ready.

e. Send the 256 triplets of random data to the MAP PIP (the data will be read back automatically, due to step d).

f. Wait until the DMA DONE flag gets set.

g. Do steps h and i 256 times, (once for each triplet of input data that was sent to the MAP).

h. From an input triplet, build a reference pair corresponding to what the Output Formatter should have produced.

i. Compare the reference pair with the actual output. If there is a difference, type two pairs of expected and received data. (Note, the heading is typed out only once).

```
ERROR IN PHASE 1
EXPECT-RECEIVE  EXPECT-RECEIVE
XXXX   XXXX     XXXX   XXXX
XXXX   XXXX     XXXX   XXXX
  :      :        :      :
  .      .        .      .
```

j. If loop on error is selected, steps b through i are repeated.

Phase 2: Verify the command Pass Formatted, 16-bit Precision. For this command, the Output Formatter expects 3 words of data input (being X, Y, and Z) and will output 4 words of data (header, X,Y, and Z).

a. Fetch 768 (=256*3) new random numbers.
b. Set the MMODE bits in the MSR to 1 (16-bit precision).
c. Send to the MAP PIP the RSR command 124000, which is for 3-D data, FSM1=0 (MOVE DRAW, MOVE DRAW,....), FSM2=5 (Pass Formatted).
d. Set up the DMA to read back 1024 (=256*4) words of data from the MAP POP whenever it gets ready.
e. Send the 256 triplets of random data to the MAP PIP (the data will be read back automatically, due to step d).
f. Wait until the DMA DONE flag gets set.
g. Do steps h and i 256 times, once for each triplet of input data that was sent to the MAP.
h. From an input triplet, build a 4-word reference set corresponding to what the Output Formatter should have produced.
i. Compare the 4-word reference set with the actual output. If there is a difference, type four pairs of expected and received data. (Note, the heading is typed out only once).

```
ERROR IN PHASE 2

EXPECT-RECEIVE.  EXPECT-RECEIVE.  EXPECT-RECEIVE.  EXPECT-RECEIVE.
XXXX   XXXX      XXXX   XXXX      XXXX   XXXX      XXXX   XXXX
XXXX   XXXX      XXXX   XXXX      XXXX   XXXX      XXXX   XXXX
  :      :         :      :         :      :         :      :
  .      .         .      .         .      .         .      .
```

j. If loop on error is selected, steps b through i are repeated.

Phase 3: Verify the command Pass, Normal Output.
For this command, the Output Formatter expects 2 words of data input (being X and Y), and will output 2 words of data (X and Y).

a. Fetch 512 (=256*2) new random numbers.
b. Set the MMODE bits in the MSR to 0 (Normal Output).
c. Send to the MAP PIP the RSR command 74000, which is for 2-D data, FSM1=0 (MOVE, DRAW, MOVE, DRAW,...) FSM2=7 (Pass).
d. Set up the DMA to read back 512 (=256*2) words of data from the MAP POP whenever it gets ready.
e. Send the 256 pairs of random data to the MAP PIP. (The data will be read back automatically, due to step d).
f. Wait until the DMA DONE flag gets set.
g. Do steps h and i 256 times, once for each pair of input data that was sent to the MAP.
h. From an input pair, build a reference pair corresponding to what the Output Formatter should have produced.
i. Compare the reference pair with the actual output. If there is a difference, type two pairs of expected and received data. (Note, the heading is typed out only once).

```
ERROR IN PHASE 3

EXPECT-RECEIVE.  EXPECT-RECEIVE.
XXXX   XXXX      XXXX   XXXX
XXXX   XXXX      XXXX   XXXX
  :      :         :      :
  .      .         .      .
```

j. If loop on error is selected, steps b through i are repeated.

Phase 4: Verify the command Pass, 16-bit Precision.
For this command, the Output Formatter expects 3 words of data input (being X, Y, and Z) and will output 4 words of data (header X, Y, and Z).

a. Fetch 768 (=256*3) new random numbers.
b. Set the MMODE bits in the MSR to 1(16-bit precision).
c. Send to the MAP PIP the RSR command 134000, which is for 3-D data, FSM1=0 (MOVE, DRAW, MOVE, DRAW,....), FSM2=7 (Pass).
d. Set up the DMA to read back 1024 (=256*4) words of data from the MPA POP whenever it gets ready.
e. Send the 256 triplets of random data to the MAP PIP (the data will be read back automatically, due to step d).
f. Wait until the DMA DONE flag gets set.
g. Do steps h and i 256 times, once for each triplet of input data that was sent to the MAP.
h. From an input triplet, build a 4-word reference set corresponding to what the Output Formatter should have produced.
i. Compare the 4-word reference set with the actual output. If there is a difference, type four pairs of expected and received data. (Note, the heading is typed out only once).

```
ERROR IN PHASE 4

EXPECT-RECEIVE.  EXPECT-RECEIVE.  EXPECT-RECEIVE.  EXPECT-RECEIVE.
XXXX   XXXX      XXXX   XXXX      XXXX   XXXX      XXXX   XXXX
XXXX   XXXX      XXXX   XXXX      XXXX   XXXX      XXXX   XXXX
  :      :         :      :         :      :         :      :
  .      .         .      .         .      .         .      .
```

j. If loop on error is selected, steps b through i are repeated.

Phase 5: Verify the command Pass Conditional, Normal Output, 'HIT'. For this command, the Output Formatter expects 2 words of data input (being X and Y), and will output 2 words of data (X and Y).

a. Fetch 512 (=256*2) new random numbers.
b. Set the MMODE bits in the MSR to 1 (16-bits precision), and set the 'HIT' and 'TAKE' bits.
c. Send to the MAP PIP the RSR command 70000, which is for 2-D data, FSM1=0 (MOVE, DRAW, MOVE, DRAW,....), FSM2=6 (Pass conditional).
d. Set up the DMA to read back 512 (=256*2) words of data from the MAP POP whenever it gets ready.
e. Send the 256 pairs of random data to the MAP PIP (the data will be read back automatically, due to step d).
f. Wait until the DMA DONE flag gets set.
g. Do steps h and i 256 times, once for each pair of input data that was sent to the MAP.
h. From an input pair, build a reference pair corresponding to what the Output Formatter should have produced.
i. Compare the reference pair with the actual output. If there is a difference, type two pairs of expected data. (Note, the heading is typed out only once).

```
ERROR IN PHASE 5

EXPECT-RECEIVE.  EXPECT-RECEIVE
XXXX   XXXX      XXXX   XXXX
XXXX   XXXX      XXXX   XXXX
  :      :         :      :
  .      .         .      .
```

j. If loop on error is selected, steps b through i are repeated.

Phase 6: Verify the command Pass conditional, 16-bit precision, 'HIT'. For this command, the Output Formatter expects 3 words of data input (being X, Y and Z) and will output 4 words of data (header X, Y and Z).

a. Fetch 768 (=256*3) new random numbers.

b. Set the MMODE bits in the MSR to 1 (16-bit precision), and set the 'HIT' and "TAKE' bits.

c. Send to the MAP PIP the RSR command 130000, which is for 3-D data, FSM1=0 (MOVE, DRAW, MOVE, DRAW,....), FSM2=6 (Pass Conditional).

d. Set up the DMA to read back 1024 (=256*4) words of data from the MAP POP whenever it gets ready.

e. Send the 256 triplets of random data to the MAP PIP (the data will be read back automatically, due to step d).

f. Wait until the DMA DONE flag gets set.

g. Do steps h and i 256 times, once for each triplet of input data that was sent to the MAP.

h. From an input triplet, build a 4-word reference set corresponding to what the Output Formatter should have produced.

i. Compare the 4-word reference set with the actual output. If there is a difference, type four pairs of expected and received data (Note, the heading is typed out only once).

ERROR IN PHASE 6

EXPECT-RECEIVE.		EXPECT-RECEIVE.		EXPECT-RECEIVE.		EXPECT-RECEIVE.	
XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX
XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX	XXXX
:	:	:	:	:	:	:	:
.	.	.	.	.	.	.	.

j. If loop on error is selected, steps b through i are repeated.

Phase 7: Verify the command Pass Conditional no 'HIT'.
For this command, the Output Formatter expects 2 words of data input (being X and Y), and will output no words of data (no 'HIT' being detected, nothing is output).

a. Fetch 512 (=256*2) new random numbers and zero the output buffer.

b. Set the MMODE bits in the MSR to 0 (Normal Output), reset the 'HIT' bit, and set the 'TAKE' bit.

c. Send to the MAP PIP the RSR command 70000, which is for 2-D data, FSM1=0 (MOVE, DRAW, MOVE, DRAW,....), FSM2=6 (Pass Conditional).

d. Set up the DMA to read back 512 (=256*2) words of data from the MAP POP whenever it gets ready.

e. Send the 256 pairs of random data to the MAP PIP (the data will be read back automatically, due to step d).

f. Wait until the MAP DONE flag gets set. Then wait for two ticks of the Real Time Clock (to allow the DMA to settle).

g. Do steps h and i 256 times, once for each possible output from the MAP (if it were broken).

h. For each possible output build a reference word of 0 to match every word in the zeroed output buffer.

i. Compare the reference of 0 with a word in the output buffer. If there is a difference, then something did come back from the MAP, which is an error.

```
ERROR IN PHASE 7

EXPECT-RECEIVE
   0    XXXX
   0    XXXX

   :     :

   .     .
```

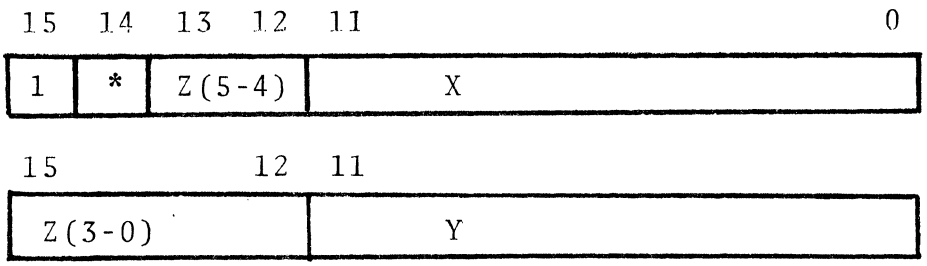
j. If loop on error is selected, steps b through i are repeated.

Error Analysis:

This section is divided into two parts: an explanation of how the MAP Output Formatter is expected to pack the input data, and a prognosis on the different types of errors that may occur.

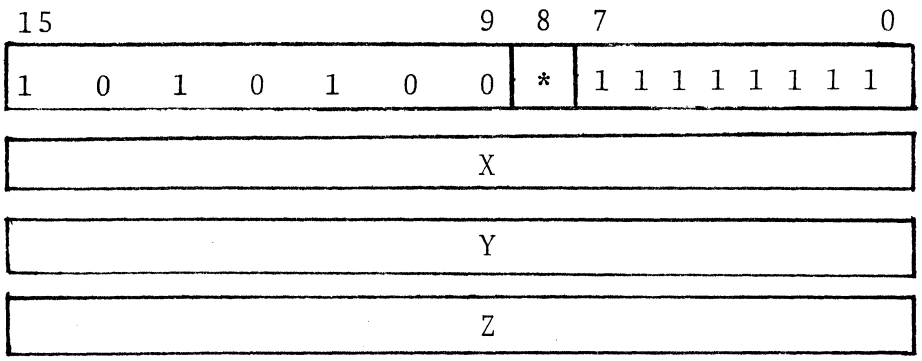
Format of Data Output by the MAP

Phase 1: Two words



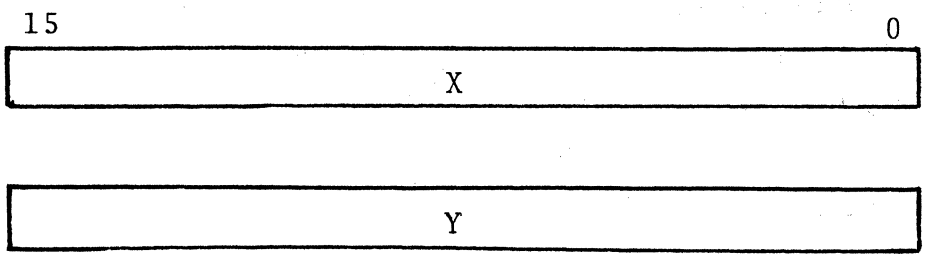
* = 0 for a MOVE
1 for a DRAW

Phase 2: Four words

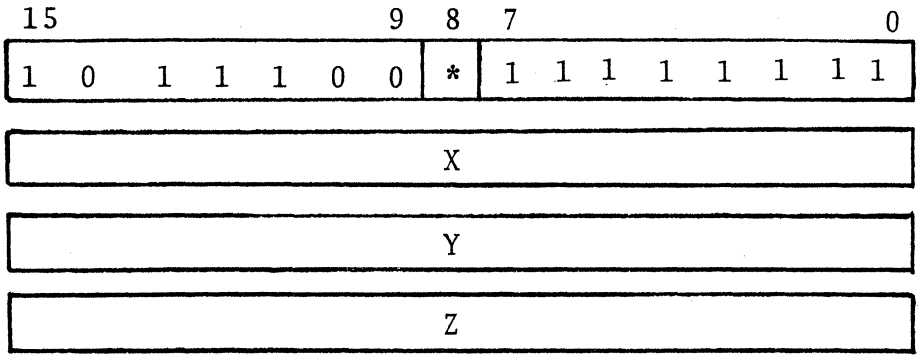


* = 0 for a MOVE
1 for a DRAW

Phase 3: Two words

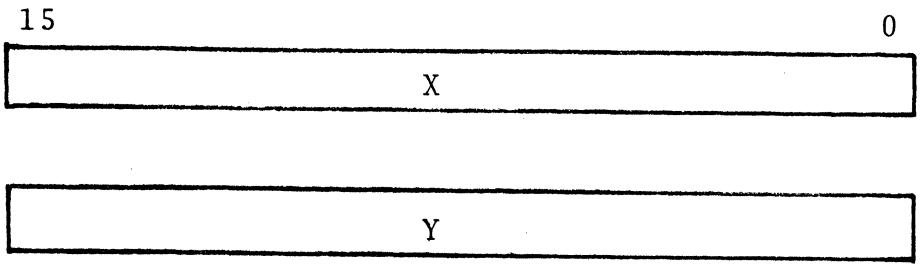


Phase 4: Four words

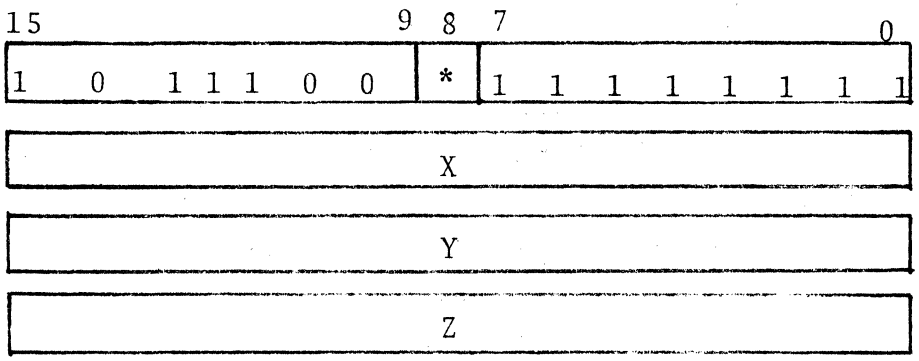


* = 0 for a MOVE
1 for a DRAW

Phase 5: Two words



Phase 6: Four words



* = 0 for a MOVE
1 for a DRAW

Phase 7: Nothing should be returned.

Error Prognosis:

1. If a single bit in a single word is wrong, suspect the 195120 card - the LS670 chips.

2. If the same bit is in error in all words, suspect the 195120 card - interconnections between chips.

3. If format is basically wrong (for instance - any output in Phase 7) suspect the 195118 card - the output controller.

4. If the test hangs (an infinite loop), check to make sure the Real-Time Clock is working.

Name: LINE GENERATOR VISUAL TEST

Functional Description:

This diagnostic provides a visual test of the Line Generator, and can be used to adjust the Line Generator (as described in the PS2 Maintenance Manual). Since the basic assumption is that the system is capable of drawing a picture, no diagnostic error messages are returned. There are 20_8 phases to this test, each running for a minimum of 5 seconds. This time may be increased by changing the refresh rate or by using the "P"ass count command. The display parameters, color(s), texture, refresh rate, etc., are set by console command during the first phase, rather than by switches. The 20_8 phases are:

1. Communication with the user.
   This phase accepts commands to: set the refresh rate, the colors, the texture, blink mode and continuous texture mode. Commands are also provided to display the available commands, display the status word being constructed, and exit phase 1.
2. 63 concentric boxes, each side of each box intensity ramped from minimum to maximum intensity.
3. 16 concentric circles.
4. 8 by 8 dot matrix.
5. Five horizontal color bars.
6. 512 lines, $1^1/_2$ inches long, covering the height of the screen.
7. Phase balance test - an "X" covering the full screen.
11. Does nothing.
12-20. Phase 2-10 are displayed by DMAing the data directly to the Picture Generator. Phases 12-20 are the same as 2-10 except that the data is deposited in PICTURE SYSTEM Memory and displayed using autorefresh mode.

NOTE: If the Real-Time Clock is broken, the test will not finish.

Program Description:

Phase 1: Build a status word to be sent to the Line Generator each phase, which contains information about color, texture, blink mode and continuous texture mode. These modes are all set as described in the PICTURE SYSTEM 2 Reference Manual, Section 2.4.3-b (Status Commands to the Line Generator). In addition, the Real-Time Clock Counts Register (RTCCNT) is set to the refresh rate desired. Cf. PS2 Reference Manual, Section 2.6.1 (Real-Time Clock).

The following sample command dialogue illustrates the use of this phase. All underlined letters are typed by the user, "(↵)" indicates "carriage return", and notes in parenthesis are further explanation of the command, as needed:

```
PHASE 2 - TYPE D,R,C,T,S,B,E, or H FOR HELP
*                    (Prompt Character)
H(↵)                 (User asking for help)
YOU MAY SET REFRESH RATE, COLOR, TEXTURE, BLINK
MODE, AND CONTINUOUS TEXTURE MODE.
THE AVAILABLE COMMANDS ARE:
E - TO EXIT
S - TO SET CONTINUOUS TEXTURE MODE
B - TO SET BLINK MODE
T - TO SET TEXTURE (WAIT FOR INSTRUCTIONS)
C - TO SET COLOR (WAIT FOR INSTRUCTIONS)
R - TO SET REFRESH RATE (WAIT FOR INSTRUCTIONS)
D - TO DISPLAY THE STATUS WORD BEING BUILT
*
D(↵)                 (Display the default status word)
200
*
S(↵)                 (Set the continuous texture bit)
*
B(↵)                 (Set the blink mode bit)
```

```
*
T (↵)              (Set the texture bits)
TEXTURE # (LESS THAN EIGHT):
6(↵)               (Long-short dashed lines)
*
C (↵)              (Set a color)
COLOR - R,Q,O,Y,G, or B:  (Note: R=red, Q=red-orange,
                                 O=orange, Y=yellow,
                                 G=green, B=black)
R (↵)              (Red)
*
C (↵)              (Set a second color)
COLOR - R,Q,O,Y,G, or B:
G (↵)              (Green)
*
D (↵)              (Display the status word)
36200              (Color is stored separately, reads
                   as 0)
*
R (↵)              (Set refresh rate)
REFRESH RATE - 120, 60, 40, 30, or 20:
30 (↵)             (Refresh rate set separately)
*
E (↵)              (Exit from phase 1)
```

The default status word and refresh rates are the system power up conditions:

Status word - 00200

DMA Refresh Rate
(Phases 2-10) - 120 hz.

Autorefresh Rate
(Phases 12-20) 60 hz.

Two types of errors may be returned:

ER1 X --- illegal command "X" encountered. Retry.

ER2 X --- illegal number "X" encountered. Retry.

Phases 2-10: Each phase runs in a two step loop for 600 clock ticks (at 60 hertz refresh rate, this means 10 sec.). The two steps are:

a. The host computer polls the PICTURE SYSTEM Real Time Clock interrupt request until a request is made.

b. It DMA's to the LG Passive Input Port, for each color requested: The status word built in phase 1 with the current color indicated, followed by the actual Line Generator data. This data instructs the Line Generator what to display. The phase descriptions which follow outline the meaning of the data sent to the LGPIP.

Phases 12-20: The same data is displayed as in tests 2-10, but in a different manner. The Picture Generator is set to autorefresh from PICTURE SYSTEM memory. For each color requested in Phase 1, the following is sent to PICTURE SYSTEM memory: The status word built with the current color indicated, followed by the actual Line Generator data. The hosts then counts the clock ticks until it can proceed to the next phase. It might be noted that the last phase displayed will stay on the screen indefinitely.

Description of Data Sent

Phases 2 & 12: Starting with the smallest square draw 63 concentric squares, and move the beam to the center at the end of the frame. The top right and bottom left corners are drawn at maximum intensity, the top left and bottom right ones at minimum. Each square is drawn in the sequence: Move to top right (maximum intensity),

draw to top left (ramp to minimum intensity), draw to bottom left (ramp to maximum intensity), draw to bottom right (ramp to minimum intensity), draw to top right (ramp to maximum intensity).

Phases 3 & 13: Starting with the smallest circle, draw 16 concentric circles, and move the beam to the center at the end of the frame. Each circle is drawn counter-clockwise from the 3 o'clock side, with 20 line segments per circle.

Phases 4 & 14: Draw 5 horizontal bands of color top one first, the top one red, then down through red-orange orange, yellow, and green for the bottom one. The speed of the beam is changed with the color as specified in the PS2 Reference Manual, Section 2.4.3-b (Line Generator Status Command). Each horizontal band of color is made up of 40 lines intensity ramped from:
minimum to maximum (left to right) for the top band.
maximum to minimum
minimum to maximum
maximum to minimum
minimum to maximum for the bottom one.
The display commands established in phase 1, with the exception of the refresh rate, will not affect this phase, since every color change must reset the status word. Move the beam to the center of the screen at the end of the frame.

Phases 5 & 15: Put the Line Generator into dot texture mode, and draw 8 rows of 8 dots, bottom to top, left to right, covering most of the screen. Since a new status command is sent out to set dot mode, all display commands from phase 1 will be ignored, with the exception of refresh rate. Move the beam to the center at the end of the frame.

Phases 6 & 16: Draw 512 horizontal lines, $1^1/_2$ inches long, on the left side of the screen. The lines are drawn bottom to top, and the frame is terminated by a move to center of screen.

Phases 7 & 17: Draw an "X" covering the entire screen in the following fashion:

Move to top right corner
Draw to bottom left
Draw to top right
Move to top left
Draw to bottom right
Draw to top left
Move to center of screen

This test is for phase adjustment of the Picture Display.

Phases 10 & 20: Draw the letters A-K and the digits 0-9 in sizes 2 through 7, in order to test the final Line Generator tuning. Move the beam to the center at the end of the frame.

Name: CHARACTER GENERATOR VISUAL TEST (Part I of III)

Functional Description:

This diagnostic provides a visual test of the Character Generator. Since the basic assumption is that the system is capable of drawing pictures, no errors are returned. There are 20_8 phases to this test, each running for 10 seconds. The length that each phase executes may be increased by setting the "P"ass count. The 20_8 phases are:

1. 'AAAA'
   This verifies that the system is capable of putting up a simple string of characters. Nulls are embedded to verify that they cause no undesirable side effects.
2. The 95 displayable characters
   The basic set of characters, from table 2.4-2 of the PICTURE SYSTEM 2 Reference Manual, are displayed in default size.
3. The format characters
   The various character positioning characters are displayed: margin settings, tabs, carriage returns, etc.
4. The 95 characters in all eight sizes.
   Allows the viewer to verify that there are no problems with any of the characters in any of the sizes. Standard character set.
5. The 95 characters, italicized, in all eight sizes.
   Allows the viewer to verify that italicizing causes no flaws in any of the characters at any size. Standard character set.
6. The 95 fast characters, in all eight sizes.
   Allows the viewer to verify that the alternate character set is complete and well-formed in all eight sizes. This character set is optimized for drawing speed.

7. The 95 fast characters, italicized, in all eight sizes Allows the viewer to verify that italicizing causes no flaws in any of the characters at any size.

10. Superscripts, Subscripts and Wraparound Test Characters at the largest size (size 7) are both superscripted and subscripted six times, to check all permissible sub-and superscripts. To verify that the character clipping works, a string of text is produced which runs off the screen. The last character on the line is expected to be whole and intact.

11-20. Phases 1-10 are displayed by DMAing the data directly to the Picture Generator. Phases 11-20 are the same as 1-10, except that the data is deposited in PICTURE SYSTEM memory and displayed using autorefresh mode.

Program Description:

Phases 1-10: Each phase runs in a two-step loop for 10 seconds. The two steps are:

a. The host computer polls the PICTURE SYSTEM Real Time Clock interrupt request (60 hz. rate), until a request is made.

b. It DMA's a set of data to the Passive Input Port of the Line Generator and goes back to wait for another interrupt request. The data instructs the Line and Character Generators what to display. The phase descriptions which follow outline the meaning of the data sent to the LGPIP.

Phases 11-20: The same data is displayed as in tests 1-10 (i.e., phases 1 and 11, 2 and 12, etc. look the same), but in a different manner. The Picture Generator is set to autorefresh PICTURE SYSTEM memory, the data is output to PICTURE SYSTEM memory and the host then waits for the length of the phase before proceeding. The last phase then, will remain on the screen indefinitely.

Phases 1 & 11: Move the beam to the center of the screen and use the ROM-defined, standard character set, at maximum intensity, and size 0. Display the characters 'AA null null AA'.

Phases 2 & 12: Move the beam to position (-3100, 1000), and use the ROM defined standard character set, at size 0 and maximum intensity. Display the characters in ASCII order from '!' to 'Z' with a few embedded spaces. Move the beam to position (-3100, 0). Display the rest of the characters in ASCII order from '[' to '~', with a few embedded spaces. The two visible lines of the expected display are the last two lines of Figure QSD027-1 but with more vertical separation between the lines.

Phases 3 & 13: Move the beam to position (-3100, 3000) and test the following format characters in order: Start of text (002), line feed (012), carriage return (015), form feed (014), backspace (010), horizontal tab (011), vertical tab (013), and cursor (036). The expected display is the same as all but the last three lines of Figure QSD027-1.

Phases 4 & 14: Move the beam to (-3614, 3770) and display eight sets of characters. Each set is in the same order as in phase 2, but at different sizes. From the top of the screen, the order is from size 0 to 7. The characters are from the standard character set. The expected display is shown in Figure QSD026-1.

Phases 5 & 15: Repeat phase 4 (or 14), but set the italics mode. The expected display is shown in Figure QSD026-2.

Phases 6 & 16: Repeat phase 4 (or 14), but use the fast character set. The expected display is shown in Figure QSD026-3.

Phases 7 & 17: Repeat phase 4 (or 14), but set the italics mode and use the fast character set. The expected display is shown in Figure QSD026-4.

Phases 10 & 20: Move the beam to (-3700, 3000) and draw the phrase 'MAX=7' at the largest size (size 7), using the standard character set. Superscript six times, drawing characters at each new superscript, ending with size 1, the smallest. Backspace and return from superscript until back at size 7 and at the end of phrase 'MAX=7'. Subscript six times, drawing characters at each new subscript, ending with size 1. Return from subscript six times, not backspacing, until back at size 7. Try to draw the word 'wraparound'. Twice the word should be terminated in the middle. Repeat the above procedure for each different character font in the order: standard, italicized standard, fast and italicized fast. The expected display is shown in Figure QSD026-5.

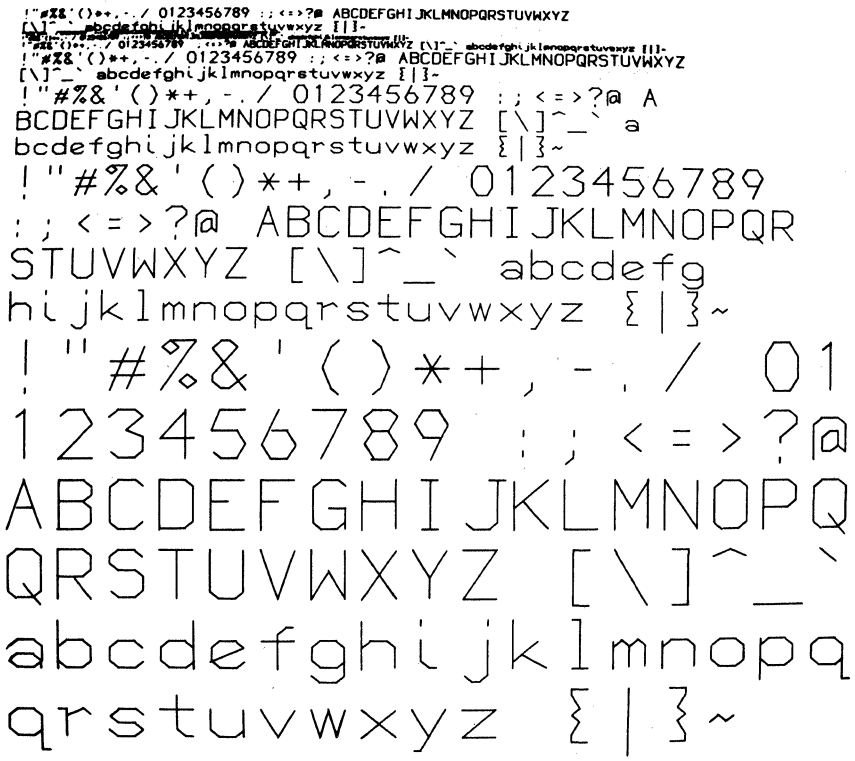


FIGURE QSD026-1

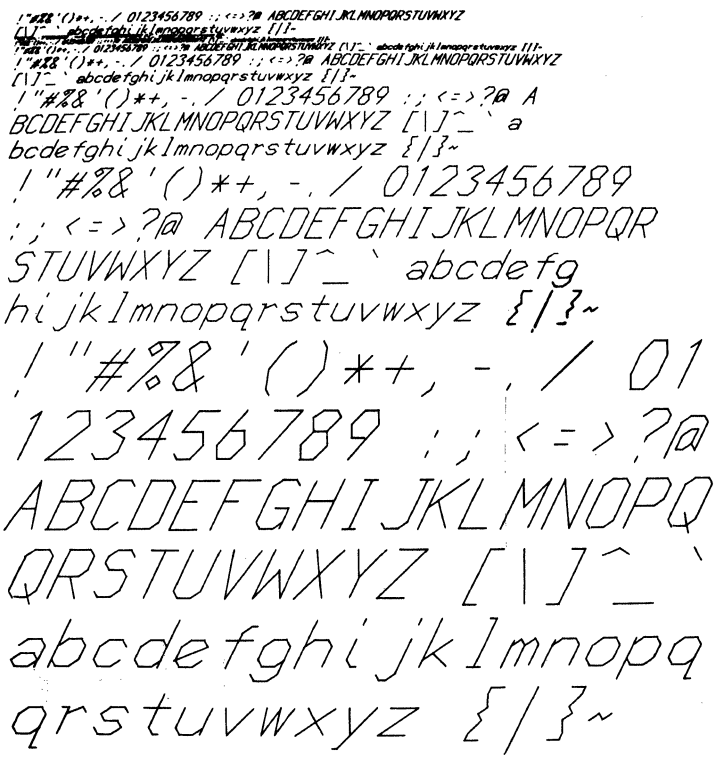


FIGURE QSD026-2

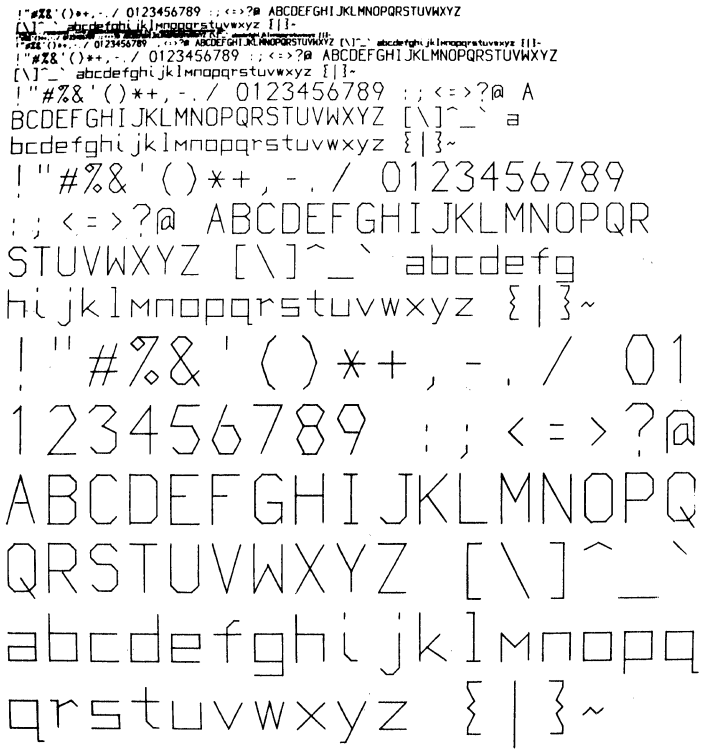


FIGURE QSD026-3

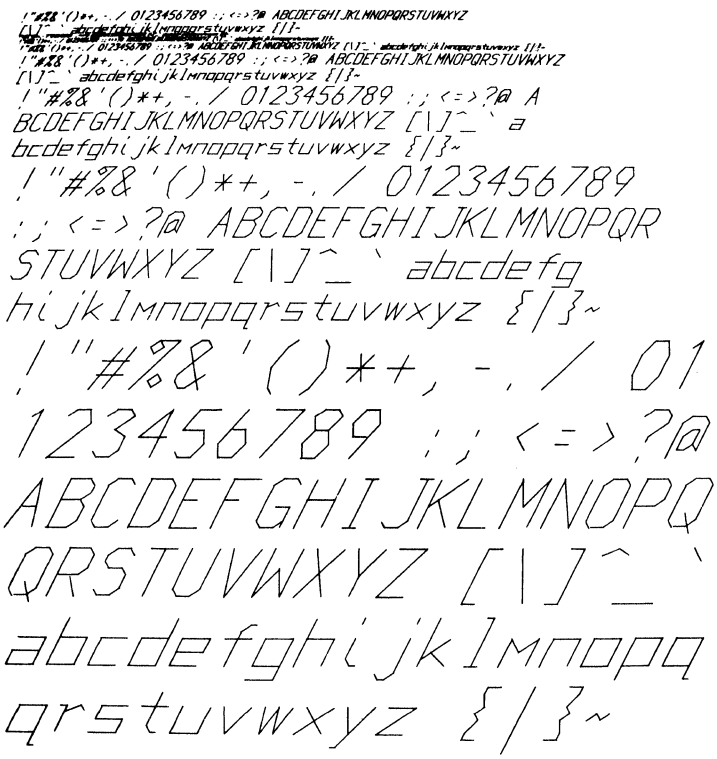


FIGURE QSD026-4

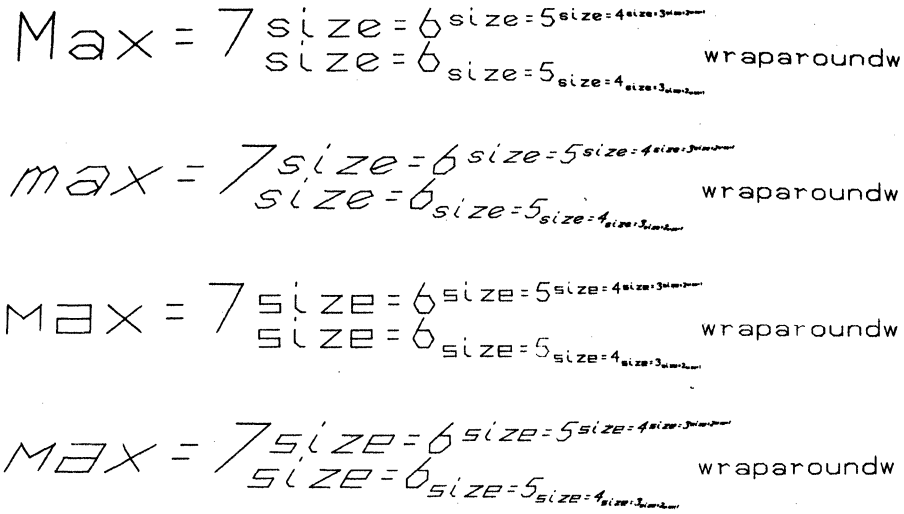


FIGURE QSD026-5

Name: CHARACTER GENERATOR VISUAL TEST (Part II of III)

Functional Description:

This diagnostic provides a visual test of the Character Generator. Since the basic assumption is that the system is capable of drawing pictures, no errors are returned. There are 15_8 phases to this test, each running for 10 seconds. The length that each phase runs may be increased by using the "P"ass count. The 15_8 phases are:

1. All 128 ASCII characters
   Those character codes which have stroke definiton (like ABC...) are displayed as such. Those which have no stroke definition (like tab, superscription, ...) are displayed by their effects - namely, tab settings, size and font changes, etc.
2. Displacement Register test
   The values in the inter-character and inter-line displacement registers can be manifested by drawing box-shaped characters. When the registers are zero, the boxes touch, otherwise they are apart according to the value of each register. Both X and Y displacements are loaded and zeroed in conjunction with the drawing of the box-shaped characters, to verify that the registers are operating.
3. Character comments
   A line of text is displayed, followed by a line of text bracketed by the command - "Do Not Display These Characters". If this command is working, only one line of text will appear on the screen, otherwise two sentences will appear.
4. Absolute Address Dispatching
   The box shaped character form phase 2 is drawn without going through the normal ASCII decoding process. Rather, the RAM address of the start of the strokes is specified, and those strokes executed directly.
5. Coefficient RAMS
   The coefficient RAMS are loaded with various size- of-character and angle combinations, to produce a display with

words at different orientations (90°, upside down, right side up).

6-10. Do nothing.

11-15. Phases 1-5 are displayed by DMAing the data directly to the Picture Generator. Phases 11-15 are the same as 1-5 except that the data is deposited in PICTURE SYSTEM memory, and displayed using autorefresh mode.

Program Description:

Previous to all phases, the character stroke RAM and the coeffiecient RAM are loaded with values as described in the PS2 Reference Manual, section 2.4.4.5-a,b "Detailed Programming of the Character Generator-Load Character Memory", and "Load Coefficient Memory". The stroke RAM is loaded with the five characters:

Square - a box, the final stroke being a "Move right and Add Displacement".

Space - a Move right and Add Displacement".

Home - like octal 002 in the ROM, to allow carriage returns.

Line Feed - a "Move down and Add Displacement".

Carriage Return - a return to the left margin defined by Home.

The coefficient RAM is loaded with 16 values: each of the four sizes 4,5,6,7 at each of the rotations 0, 90, 180, and 270 degrees.

Phases 1-5: Each phase runs in a two-step loop for 10 seconds. The two steps are:

a. The host computer polls the PICTURE SYSTEM Real Time Clock interrupt request (60 hz. rate), until a request is made.

b. It DMA's a set of data to the Passive Input Port of the Line Generator and goes back to wait for another interrupt request. The data instructs the Line and Character Generators what to display. The phase descriptions which follow outline the meaning of the data sent to the LGPIP.

Phases 6-10: Do nothing.

Phases 11-15: The same data is displayed as in tests 1-5 9i.e., phases 1 and 11, 2 and 12 etc. look the same), but in a different manner. The Picture Generator is set to autorefresh PICTURE SYSTEM memory, the data is sent to the PICTURE SYSTEM memory, and the host then waits for the length of the phase before proceeding. The last phase, then will remain on the screen indefinitely.

Phases 1 & 11: Move the beam to position (-3700, 3000) and test the following format characters in order:

Start of text (002), line feed (012), carriage return (015), form feed (014), backspace (010), horizontal tab (011), vertical tab (013), and cursor (036).

The following control characters in order: a-hat (001), push load font parameter stack (003), followed by a character size specification and a character, pop font parameter stack (004), e-hat (005), f-hat (006), g-hat (007), n-hat (016), o-hat (017), p-hat (020), q-hat (021), r-hat (022), s-hat (023), t-hat (024), u-hat (025), left angle bracket (026), right angle bracket (027), set superscript (030) - digit followed by set superscript followed by digit, reset superscript (031), set subscript (032) - digit followed by set subscript followed by digit, reset subscript (033), set italics (034) - followed by character, reset italics (035) - followed by character, blinking cursor (036), o-hat (037).

The basic displayable characters in order from "!" to "~".

The display should look like figure QSD027-1.

Phases 2 & 12: Test the X and Y displacement register. Cf the PS2 Reference Manual, Section 2.4.4.5-c "Detailed Programming of the Character Gneerator - Load Inter-Character and Inter-Line Spacing Registers". All characters will be executed from the stroke RAM, (the box is the only visible character).

a. Move beam to position (0,0).
b. Zero displacement registers (X and Y).
c. Draw six boxes (sides should touch), carriage return.
d. X-register gets 1/4 box width, Y-register gets zero.
e. Draw five boxes (tops should touch row above, left side of line of five boxes and right side of line of five boxes should exactly match the sides of the line above).
f. X-register gets 1/4 box width, Y-register gets non-zero value.
g. Carriage Return X-register gets 1/4 box width, Y-register gets zero.
h. Draw five boxes (tops should not touch row above, but otherwise be identical to the line of five above).
i. Zero both registers.
j. Draw six boxes (tops should touch row above, sides should touch as in top row, left and right sides of line should line up with line above).

Note that the contents of the X-register (measured in screen units) are calibrated against the Line Generator in the X-direction. If the rows don't all line up on their left and right sides, then either the X-register has bad data, or the Line Generator has a fault.
The display should look like figure QSD027-2.

Phases 3 & 13: Move beam to position (-3700, 0).
Display the phrase:

YOU SHOULD ONLY SEE THIS LINE OF TEXT.

Set the "Comment Characters" state (i.e., do not display), and send down the phrase:

THE TRANSFER FUNCTION (OCTAL 26,42) IS BROKEN.

If the last phrase appears on the screen, then the Character Generator is indeed broken. The sequence bytes 26,42 should stop the display of all following characters until the byte 27. Cf PS2 Reference Manual, Section 2.4.4.5-d "Detailed Programming of the Character Generator - comments within character strings".

Phases 4 & 14: Move beam to position (0.0).
Send to the Character Generator the sequence of bytes: (26, 104, 120, 105, 27).
This should start drawing the contents of the character stroke RAM from location 5. The box character from phase 2, should appear in the center of the screen. Cf. the PS2 Reference Manual, Section 2.4.4.5-e "Detailed Programming of the Character Generator - Direct Dispatch to Character Memory Location".

Phases 5 & 15: The contents of the coefficient RAM are tested.
In all four phrases displayed, the first letter of each word is one size larger than the rest. The four phrases are:

(A) EVANS & SUTHERLAND
(B) PICTURE SYSTEM
(C) COMPUTER GRAPHICS
(D) TEXT TERMINAL

The phrases are drawn as defining two boxes, one surrounding the whole screen, one surrounding the inside quarter. The following table gives the position and orientation of the phrases (all angles

go counter-clockwise):

	Start Position	Angle (degrees)	Phrase
LARGE BOX:	Bottom Left	0	(A)
	Bottom Right	90	(B)
	Top Right	180	(C)
	Top Left	270	(D)
SMALL BOX:	Top Left	0	(A)
	Top Right	270	(B)
	Bottom Right	180	(C)
	Bottom Left	90	(D)

The display should like figure QSD027-3.

```
0*THIS IS MY TOP LEFT MARGIN SETTING (START OF TEXT)
                                                        1* THIS IS ONE LINE DOWN (LI
2* THIS IS ONE LINE DOWN, ON THE LEFT MARGIN (LINE FEED, CARRIAGE RETURN)
3*THIS SHOULD BE LINE 3 ON THE LEFT MARGIN (FORM FEED, 3 LINE FEEDS)
4* THE UNDERLINE VERIFIES THAT BACKSPACE WORKS
5*THESE *TAB...  *STOPS.  *ARE...  *ALWAYS *SET...  *TO....  *EIGHT.  *SPACES *APART
6*       8        16       24       32       40       48       56       64       72
7*
8* IF THIS IS LINE 8, THEN VERTICAL TAB WORKS ("8", FORM FEED, VERTICAL TAB, " *
9* A CURSOR SHOULD BLINK UNDER THE ^

 A  efgnopqrstu<>4343 B

!"#%%&()*+,-./ 0123456789 :;<=>?@ ABCDEFGHIJKLMNOPQRSTUVWXYZ
[\]^_` abcdefghijklmnopqrstuvwxyz {|}~
```

FIGURE QSD027-1

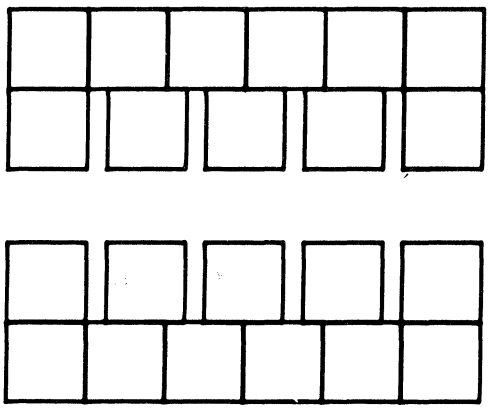


FIGURE QSD027-2

FIGURE QSD027-3

Name: CHARACTER GENERATOR VISUAL TEST (Part III of III)

Functional Description:

This test has three purposes:

1. Visual verification of the 128 standard character definitions in the character ROM of the Character Generator.
2. Visual test of the Character Generator character RAM memory.
3. Demonstration of the capability for non-standard character definitions in the character RAM.

The test consists of four phases: 1, 2, 11, and 12.

Phase 1 transmits text data, described below, to the Line Generator. The text is drawn in two fonts:

1. ROM normal font
2. ROM fast font

Differences may be observed in the lower case letters of the two fonts. The text is transmitted by repeated DMA transfers.

Phase 2 begins by loading the character RAM with character definitions identical to those in the ROM, with the exception of fast (lower case) a, b, c, d, e, f, and g. The fast "g" is defined as a no-op. The other characters are redefined as ▷◁↑↓△ respectively. The text data of phase 1 is again transmitted by repeated DMA transfers, to be drawn in RAM normal font, and RAM fast font. The visual effect of phase 2 should be identical to that of phase 1, except that fast a-g will be replaced by the non-standard text above. Any other discrepancies indicate a failure of the character RAM memory.

The text sequence for all four phases is as follows, where numbers with leading zero represent octal ASCII codes:

1. Line Generator status = absolute, Character Generator reset.
2. Move to -3700, 3000, 77.

3. Push fast ROM/RAM font.
4. Load normal ROM/RAM font.
5. Control Character Sequence: 02,013,[VT]ABC,012,[LF], 015,[CR],012,0114,0106,040,01,05-07,016-027,015,012, AB,010,BACKSPACE,011,[HT],012,[LF],015,[CR],012,ABC, 030,SUP,031,RSP,032,SUB,033,RSB,034,ITALICS,035,RIT 036,CURSOR,037,014,[FF],015,012
6. Printing Alphabet: !"#$%&'()*+,-./0123456789:;<= >?@,015,012,A-Z.[ ]^,a-z,{|}~
7. 04,[PFT],013,013
8. Control Character Sequence (cf 5; fast font).
9. Printing Alphabet: (cf 6; fast font).

Phase 11 causes the data sequence of phase 1 to be transmitted to the Character Generator by auto refresh, rather than by DMA. The refresh buffer begins in location 0 of PICTURE SYSTEM memory. Phase 12 loads the character RAM in the same manner as phase 2 (by DMA), but then causes the text data of phase 2 to be transmitted by auto refresh, rather than by DMA.

Each phase has a duration of 10 seconds per pass. Phases 11 and 12 leave auto refresh in operation when they terminate.

Program Description:

Phase 1: Display the full character set in ROM normal font and ROM fast font, by DMA. The program steps of Phase 1 are the steps of Phase 2, except that step (c) is omitted.

Phase 2: Display the full character set in RAM normal font and RAM fast font, by DMA.

a. Output message, and reset the Picture system.
b. Set real-time clock counts = 60 HZ.
c. Load the Character RAM.

d. Load the data buffer DBUF with the text data described above in the functional description.
e. Repeat (f) and (g) 600 times for each pass.
f. Wait for real-time clock.
g. Transmit DBUF to the Line Generator passive input port by DMA.
h. End of phase.

Phase 11: Display the full character set in ROM normal font and ROM fast font, by auto refresh. The program steps of Phase 11 are those of Phase 12, except that step (c) is omitted.

Phase 12: Display the full character set in RAM normal font and RAM fast font, by auto refresh.

a. Output message, and reset the Picture System.
b. Set real-time clock counts = 60 HZ.
c. Load the character RAM.
d. Load the data buffer DBUF with the text data described above in the functional description.
e. For each pass, repeat (f) through (h).
f. Transmit DBUF to PSMEM beginning at location 0, by DMA.
g. Set Refresh Active Start Address, Refresh Active Input Limit, and start auto refresh.
h. Wait for 600 real-time clock counts.
i. End of phase.

Error Analysis:

Although this test outputs no error messages, the displayed text may be inspected to determine whether the character ROM and RAM are functioning properly.

Phases 1 and 11 demonstrate, by the absence of abnormality in the display, that the Character ROM is being properly read by the character generator. The Character Stroke RAM and ROM

are on the 195222-100 card. This card is the most likely suspect if the characters are in error. The 195223-100 is the control for the character generator, with the 195221-100 containing the character size and rotation control.

The display of phase 2 (12) should be identical to that of phase 1 (11) except that the fast lower case a through g are replaced with the non-standard character definitions described above. If phase 2 fails (but phase 1 passes), the Character Stroke RAM (or its addressing) on the 195222-100 card is the most likely suspect.

Name: REFRESH CONTROLLER CONTROL & DATA PATH TEST

Functional Description:

This diagnostic is used to test registers and data paths within the refresh controller. It consists of seven phases:

1. Refresh Status Register Test
   Checks that all registers can be cleared, checks bit patterns in status register, checks that reset initializes status register, checks that IE bits set and are cleared by RESET, checks that request bits don't set by clearing.
2. Refresh Controller Registers Test
   Checks registers other than status register for ones, zeros, checkerboard. Checks unique addressing.
3. Bus and Request Test
   Checks that RFSTOP request can be set. Checks that it can be cleared by writing and by PGRESET.
4. Internal Data Path Test
   Checks that internal registers can be loaded and read back. Checks that ones, zeros and checkerboard patterns can be internally transmitted from RFASA to RFAIA and that other registers are undisturbed. Checks comparator by RFAIA, RFAIL equality and "equality plus one".
5. Data Path and Writeback Control Test
   Checks that RFHOLD logic stops refresh controller when RFAWA and RFAWL are equal or within one of being equal for various zero, one and checkerboard addresses.
6. Writeback Data Test
   Checks that data can be relocated within PSMEMORY in writeback mode. Also checks that PGRESET will clear writeback request bit.

7. Segment Match Test

Checks match logic for any name, right, left and full name match. Checks MATCH HOLD and MATCH REQUEST logic. Checks segment jump capability.

Program Description:

Phase 1: The seven control registers are zeroed.
If an error occurs, the following message is output.

1: RF REG ZERO ERR; ADDR=XXXXX EXPT=XXXXX RECD=XXXXX INDEX=22

The status register is tested to verify all READ/WRITE bits work.
If an error occurs, the following message is output.

2: RFSR BIT SET/CLEAR ERR; ADDR=XXXXX EXPT=XXXXX RECD=XXXXX INDEX=44

PGRESET is tested to see if it will reset the status register and interrupt enable bits properly.
If an error occurs, the following message is output.

3: PG RESET TEST; ADDR=XXXXX EXPT=XXXXX RECD=XXXXX INDEX=62, 70 or 75

Request bits are tested to see that they can't be directly set by software.
If an error occurs the following message is output.

3: PG RESET TEST; ADDR=XXXXX EXPT=XXXXX RECD=XXXXX INDEX=101, 104 or 106

Phase 2: The seven control registers are checked with zeros, ones and checkerboard patterns.
If an error occurs the following message is output.

5: RF REG BIT ERR; ADDR=XXXXX EXPT=XXXXX RECD=XXXXX INDEX=32

The seven control registers are checked for unique addressing by zeroing all, loading one and checking all.
If an error occurs, the following message is output.

6: RF UNIQ ADDR ERR; ADDR=XXXXX EXPT=XXXXX RECD=XXXXX INDEX=102

Phase 3: The stopped logic and request flip flop are tested to see that RFSTOPREQ can be set and cleared. The seven control registers are cleared. The RFSTART bit is set. The RFSTOPPED bit is checked.
If an error occurs, the following message is output.

7: RF REQ TST ERR; ADDR=XXXXX EXPT=XXXXX RECD=XXXXX INDEX=36

The RFSTOPREQ bit is checked.
If an error occurs, the following message is output.

7: RF REQ TST ERR; ADDR=XXXXX EXPT=XXXXX RECD=XXXXX INDEX=44

The RFSTOPREQ is cleared by writing a one into it.
If an error occurs, the following message is output.

7: RF REQ TST ERR; ADDR=XXXXX EXPT=XXXXX RECD=XXXXX INDEX=47

A similar sequence is performed but RFSTOPREQ is cleared by PGRESET.
If an error occurs, the following message is output.

7: RF REQ TST ERR; ADDR=XXXXX EXPT=XXXXX RECD=XXXXX INDEX=60, 65 or 76 respectively.

Phase 4: Internal data paths are tested by transferring zero, one and checkerboard patterns from the RFASA to the RFAIA registers. The address comparator is tested by the same patterns to detect equality and "equality plus one".
If an error occurs, the following message is output.

10: RF DATA PATH ERR; ADDR=XXXXX EXPT=XXXXX RECD=XXXXX INDEX=53

Failure of RFAIA to hold desired contents is indicated by message.

10: RF DATA PATH ERR; ADDR=XXXXX EXPT=XXXXX RECD=XXXXX INDEX=56

Failure of other registers to remain zero is indicated by message.

10: RF DATA PATH ERR; ADDR=XXXXX EXPT=XXXXX RECD=XXXXX INDEX=67

Phase 5: Comparator is tested by loading RFAWA and RFAWL with zero, one and checkerboard patterns to verify that equality and "equality plus one" will stop refresh controller when writeback is set.
If an error occurs, the following message is output.

11: RF WB CTL ERR; ADDR=XXXXX EXPT=XXXXX RECD=XXXXX INDEX=70

Failure of RFAIA to remain the same is indicated by message.

11: RF WB CTL ERR; ADDR=XXXXX EXPT=XXXXX RECD=XXXXX INDEX=75

WBSTOP REQ only is tested to be set.
Failure to set or additional request bits setting is indicated by message.

11: RF WB CTL ERR; ADDR=XXXXX EXPT=XXXXX RECD=XXXXX INDEX=101

Writeback is cleared and WBSTOPREQ is tested to remain set.
Failure is indicated by message.

11: RF WB CTL ERR; ADDR=XXXXX EXPT=XXXXX RECD=XXXXX INDEX=115

WBSTOPREQ is acknowledged and tested to be clear.
Failure is indicated by message.

11: RF WB CTL ERR; ADDR=XXXXX EXPT=XXXXX RECD=XXXXX INDEX=123

Phase 6: PS memory locations 200 to 221 are loaded with a status command, a status command, a MOVETO command, 3 character commands, a MOVETO command and 3 different character commands. It is assumed that PICTURE SYSTEM memory. functions correctly. Refresh with writeback is started. The RFAWA is tested to see that transfer is complete. If an error occurs, the following message is output.

12: RF WB DATA ERR; ADDR=XXXXX EXPT=XXXXX RECD=XXXXX INDEX=50

The first set of character data should be written over the second set. (The characters RFDATATEST are written over XXXXXXXXXX).
If an error occurs, the following message is output.

12: RF WB DATA ERR; ADDR=XXX EXPT=XXXXX RECD=XXXXX INDEX=63

The writeback bit in the RF status register is cleared and WBSTOPREQ is checked to remain set.
If an error occurs, the following message is output.

12: RF WB DATA ERR; ADDR=XXXXX EXPT=XXXXX RECD=XXXXX INDEX=76

PGRESET is issued and WBSTOPREQ is checked to be cleared.
If an error occurs, the following message is output.

12: RF WB DATA ERR; ADDR=XXXXX EXPT=XXXXX RECD=XXXXX INDEX=105

Phase 7: PS Memory is loaded, starting at location 200 with a segment halt, a status command, eight sets of various segment names each followed by a null character command, and a segment jump back to the segment halt. The Refresh Controller is set to SEARCH, MODE 0, MATCH HOLD and is started at location 202 (the status command). The RFHOLD bit in the status register is tested. If an error occurs, the following message is output.

13: RF MATCH ERR; ADDR=XXXXX EXPT=XXXXX RECD=XXXXX INDEX=47

The MATCH REQ bit is tested.
If an error occurs, the following message is output.

13: RF MATCH ERR; ADDR=XXXXX EXPT=XXXXX RECD=XXXXX INDEX=54

The RFAIA is tested to see if it points just after the first segment name at location 206.
If an error occurs, the following message is output.

13: RF MATCH ERR; ADDR=XXXXXX EXPT=206 RECD=XXXXX INDEX=61

SEARCH, MODE 1 and MATCH HOLD are set and MATCH REQ is cleared to cause the refresh controller to continue. RFAIA is tested to have stopped at 216.
If an error occurs, the following message is output.

13: RF MATCH ERR; ADDR=XXXXX EXPT=216 RECD=XXXXX INDEX=76

SEARCH, MODE 2 and MATCH HOLD are set and refresh is continued. RFAIA is tested to be 226.
If an error occurs, the following message is output.

13: RF MATCH ERR; ADDR=XXXXX EXPT=226 RECD=XXXXX INDEX=110

SEARCH, MODE 3, MATCH HOLD are set and refresh is continued. RFAIA is tested to be 236.
If an error occurs, the following message is output.

13: RF MATCH ERR; ADDR=XXXXX EXPT=236 RECD=XXXXX INDEX=122

SEARCH, MODE 0, MATCH HOLD, MATCH DEC are set and refresh is continued. RFAIA is tested to be 240.
If an error occurs, the following message is output.

13: RF MATCH ERR; ADDR=XXXXX EXPT=240 RECD=XXXXX INDEX=134

SEARCH MODE and MATCH HOLD, MATCH DEC are cleared and refresh is continued. The RFSTOPPED bit is tested.
If an error occurs, the following message is output.

13: RF MATCH ERR; ADDR=XXXXX EXPT=100000 RECD=XXXXX INDEX=147

RFAIA is tested to be 202.
If an error occurs, the following message is output.

13: RF MATCH ERR; ADDR=XXXXX EXPT=202 RECD=XXXXX INDEX=154

Error Analysis:

Error #	Prognosis
1	This error indicates that the refresh control registers (74S172 IC's on the 195153 card) cannot write and/or read zeros. This could be caused by the registers or their addresses which are generated on the 195151-100 (Sheet 2). If the dip switches are not set properly on the 195151-100 card, an error will occur.
2	This error indicates that the status bits could not be read or written. The Status Register is located on Sheet 3 of the 195152-100 card.
3, Index = 62, 70, or 75	This error indicates that the status bits could not be reset to zero by PSRESET. PSRESET is generated by the 195218-100 in the line generator.
3, Index = 101, 104, or 106	This error indicates that the interrupt request bits on sheet 3 of the 195151-100 card are being set (instead of cleared).
5	This error indicates a data path error in the refresh control registers. Check the 8T26 or 74S172 on the 195153-100 card. (Sheets 2, 3, or 4).
6	This error indicates an addressing error of the refresh control registers on the 195153-100 card. The address decode logic exists on sheet 2 of the 195151-100 card.

Error Analysis:

Error #	Prognosis
7 Index = 36	This error indicates an error may exist in the Detector/arbitrator logic on the 195151-100 card (Sheet 4). Also a problem may exist in the State Machine on the 195152-100 (Sheet 2). The detailed state flow diagram is contained in the maintenance manual for the Refresh Controller.
7 Index = 44	This error indicates that setting RF STOPPED didn't cause RF STOPREQ bit to set. Check the Interrupt logic on 195151-100 (Sheet 3). The hardware name of RF STOPREQ is HLTREQ.
7 Index = 47	This error indicates that RF STOPREQ (HLTREQ) couldn't be cleared. Check the Interrupt logic on 195151-100 card (Sheet 3).
7 60, 65, 76	This error indicates that RF STOPREQ (HLTREQ) couldn't be cleared by PGRESET. Check Interrupt logic on 195151-100 card (Sheet 3).
10 Index = 53	This error indicates a problem on the TBUS and VBUS and the address. Check the 74S283, 74S172 IC's on the 195153-100 card.
10 Index = 56	This error indicates that address didn't increment Check 74S283 on 195153-100 card.
10 Index = 67	This error in icates an addressing problem. Check 74S172 and State Machine on the 195152-100 card (Sheet 2). The addresses generated by the detailed state flow diagram is contained in the maintenance manual for the Refresh Controller.

Error Analysis:

Error	Prognosis
11 Index = 70	This error indicates that the comparator is bad on the 195153-100 card (Sheet 4).
11 Index = 75	This error indicates that the comparator was unable to stop the RF control (i.e. the state machine did not see the match condition). The HLTREQ and MATREQ Flip-Flops on 195151-100 (Sheet 3) should have been set. If they are set, then the state machine on the 195152-100 (Sheet 2) did not respond to the HLTREQ flip-flop.
11 Index = 101	This error indicates that the WBSTOPREQ (WBHREQ) bit cannot be set. Check interrupt logic on 195151-100 card (Sheet 3).
11 Index = 115	This error indicates that the WBSTOPREQ (WBHREQ) was cleared, (should remain set). Check Interrupt logic on 195151-100 card (Sheet 3).
11 Index = 123	This error indicates that WBSTOPREQ (WBHREQ) couldn't be cleared by program control Check Interrupt logic on 195151-100 card (Sheet 3).
12 Index = 50	This error indicates that the adder is not working properly. Check 195153-100 card (Sheet 4).
12 Index = 63	This error indicates that the data is not being transferred correctly. Check the PSA register on the 195153-100 (Sheet 4) for the value. Also this test is using the writeback mode in the memory system 195140-100 series cards which could also be at fault.

Error Analysis:

Error	Prognosis
12 Index = 76	This error indicates that WBSTOPREQ (WBHREQ) cleared (should remain set). Check Interrupt control on 195151-100 card (Sheet 3).
12 Index = 105	This error indicates that WBSTOPREQ (WBHREQ) couldn't be cleared by PGRESET. Check Interrupt control on 195151-100 card (Sheet 3).
13 Index = 47	This error indicates that the Name Match didn't take place (i.e. the state machine did not see the match condition). The HLTREQ and MATREQ Flip-Flops on 195151-100 (Sheet 3) should have been set. If they are set, then the state machine on the 195152-100 (Sheet 2) did not respond to the HLTREQ flip-flop.
13 Index = 54	This error indicates that MATCHREQ (MATREQ) was set by Match. Check Interrupt logic on 195151-100 card (Sheet 3).
13 Index = 61	This error indicates that the RF controller found the wrong name. Check comparator on the 195153-100 card (Sheet 4).
13 Index = 76, 110, 122	These errors are the same as 7C except for the Search Mode. Check Comparator on 195153-100 card (Sheet 4) and Interrupt logics on 195151-100 card (Sheet 3).
13 Index = 134	This error indicates that the Adder can't Subtract. Check the adder on 195153-100 card (Sheet 4).

Error Analysis:

Error	Prognosis
13 Index = 147	This error indicates that the Refresh Controller didn't stop. This means that the STATUS HALT or the RFJUMP didn't work. Check control logic on 195152-100 card (Sheet 2).
13 Index = 154	This error indicates that Segment Jump didn't work. Check control logic on 195152-100 card (Sheet 2).

NAME: Character Generator RAM Test

Functional Description:

QSD033 tests the Character and Coefficient RAM's in the Character Generator. It assumes that test QSD032 passes. Each RAM is tested with three patterns: a checkerboard, an inverse checkerboard, and an address-data pattern. There are two phases:

1. Test the Character RAM with all three patterns.
2. Test the Coefficient RAM with all three patterns.

Program Description

Phase 1: Character RAM Test

a. Do a PS Reset
b. Form an address-data pattern in a 1024 word buffer. Set word 0=1, word 1=2, ..., word 62=63, then start over with word 63=1, word 64=2. etc. Keep counting up to 63 until all 1024 words are filled. The purpose of this pattern is to discover if any address lines are tied together. Hence the count must go up to an odd number, like 63, and then restart.
c. Format the data for the Character Generator: Keep bits 5-0 intact, set bit 6, clear bit 7, and copy bits 7-0 into bits 15-8. Do this for the entire word buffer.
d. Do a PS Reset.
e. Send the data (240,0) to the PGPIP. This will set the PFORM CCHAR bit.
f. Send the transfer function for load Character Memory to the PGPIP (bytes 26, 113, 160, 100), followed by the 1024 word buffer (to load the entire character RAM) followed by bytes 27,0,0,0, (to terminate the function). Send via the DMA, and don't wait for the transfer to finish.

g. Wait for the DMA transfer to finish. If it has not finished in 1024 timeouts, type:

RAM WRITE FAILED, RUN ANOTHER DIAGNOSTIC

NOTE: This error message means that something is wrong with either the DMA function or the Character Generator load character memory function. Other diagnostics test these better than this one does.

h. Send a Read Character Memory transfer function (bytes 26, 115, 120, 100, 27, 0, 0, 0) to the PGPIP.

i. The Character Generator should be in state 176. If it isn't, single step it in maintenance mode as many as 1024 times, until it reaches state 176. If it still never gets there, type:

RAM READ FAILED, RUN ANOTHER DIAGNOSTIC

NOTE: This means that something is probably wrong with the Read Character Memory sequence, which should be tested with QSD032.

j. Single step it 2 more times in maintenance mode.

k. For all 1024 words in the memory, repeat steps 1-0.

l. Read the character memory.

m. Form the expected data: take bits 5-0 from the word in the original data buffer and put them into bits 5-0 of of the expected data word. Take bits 13-8 from the word in the original data buffer and put them into bits 11-6 of the expected data word. For the two checkerboard tests, this will yield a 12-bit checkerboard pattern.

n. Compare the received data word against the calculated expected data word. If they are different, type:

STROKE RAM ERR; LOC=XXX EXPT=XXX RECD=XXX

where: LOC - is the address that failed
EXPT- is the expected data
RECD - is the received data

NOTE: When the "L"oop on error 1 option is selected, the test will go back to step (e) and repeat, stepping directly up to the failing location, which it will again read and check.

o. Single step the Picture Generator in maintenance mode.

p. After steps 1-o have been done for all 1023 words of the RAM, let the machine free-run again, to exit the read function.

---

q. Repeat steps c-p, only generate checkerboard data in the 1024 word buffer. That is, let word 0=125252, word 1=052525, word 2=125252, etc.

r. Repeat steps c-p, using a reverse checkerboard pattern. That is, word 0=052525, word 1=125252, etc.

Phase 2: Coefficient RAM Tests

a. Do a PS Reset.

b. Put an address-data pattern into the 1024 word buffer exactly as in (b) of Phase 1.

c. Repeat steps d-h 128 times (the Coefficient RAMs only accept 32 sets of 4 words each, or 64 words each for X and Y).

d. Copy the next word from the 1024 word buffer into a 128 word buffer, to be used later for the expected data calculation.

e. If bit 6 of the data word is set, set the upper byte of the word to be sent equal to 103, otherwise set it to 100.

f. Leave bits 5-0 alone, set bit 6, and clear bit 7 of the word to be sent.

g. If this is not a PDP-11 type machine (with backwords bytes), swap the upper and lower bytes.
h. Put the word to be sent back into the 1024 word buffer from where it came. It is now formatted.
i. Do a PS Reset.
j. Send a PFORM CCHAR for status command to the PGPIP.
k. DMA to the PGPIP a Load Coefficient Memory function (bytes 26, 127, 140, 101), the first 128 words of the 1024 word buffer, and an end transfer function (bytes 27, 0, 0, 0,).
l. Give the Character Generator 1024 timeouts to finish the DMA. If it still has not finished, type:

RAM WRITE FAILED, RUN ANOTHER DIAGNOSTIC

NOTE: This error message means that something is wrong with either the DMA function or the Load Coefficient Memory function. Other diagnostics should be used to diagnose these problems.

m. Send a Read Coefficient Memory function to the PGPIP. (bytes 26, 135, 140, 101, 100, 120, 27, 0).
n. The Character Generator should be in state 162. If it isn't, single step it in Maintenance Mode up to 1024 times, until it does reach state 162. If it never reaches that state, type:

RAM READ FAILED, RUN ANOTHER DIAGNOSTIC

NOTE: This means that something is wrong with the Read Coefficient Memory function. QSD032 tests this function.

o. Single step it 3 more times in Maintenance Mode.

p. The A and C values are all stored in the X-Coefficient RAM, so that location 0-63 hold the coefficients 0A, 0C, 1A, 1C, ..., 31A, 31C. For all 64 values, repeat steps q-t.

q. Read the Coefficient Memory.

r. Form the expected data from the 128 word buffer established in step (d): Keep bits 6-0 the same. If bit 6 is set, set also bits 11-7, otherwise reset them. Bits 15-12 are 0.

s. Compare the received and expected data. IF they are different, type:

COEFF.RAM ERR; LOC=XXX EXPT=XXX RECD=XXX

where: LOC - is the failing location (0-63)
EXPT - is the expected data
RECD - is the received data

NOTE: When "L"oop on error 2 is specified, the test will go back to step (i), and repeat, stepping directly up to the failing location, which it will again read and check.

t. Single step the Character Generator twice in Maintenance Mode.

u. After steps q-t have been done for all 63 X-values, free run the Character Generator to end the transfer function.

---

v. Repeat steps m-u, but sending instead a Read Coefficient Memory, B & D, transfer function (bytes 26, 135, 140, 101, 100, 101, 270). This reads the 64 values in the Y-RAM (0B, 0D, 1B, 1D, ..., 31B, 31D).

w. Repeat steps c-v, only put a checkerboard pattern in the 1024 word buffer to start with. That is, word 0=125252, word 1=052525, etc.

x. Repeat steps c-v, putting a reverse checkerboard pattern into the 1024 word buffer. That is, set word 0=052525, word 1=125252, etc.

Error Analysis:

Since it is assumed that QSD032 passes, the only new things being checked are the RAM's.

If the STROKE RAM fails, look to chips 38-49 on the 19522 card.

If the COEFFICIENT RAM fails, look to chips 41 and 49 on the 194221 card.

Identification: QSD100
Date: 12/16/76

M H2 = XXX Sets desired PDP-11 vector address

Name: PS2 INTERRUPT DIAGNOSTICS

Functional Description:

The diagnostic is used to verify the following:

a. That PICTURE SYSTEM interrupts (CLOCK, SYSTEM and DEVICE) can be transmitted to and processed by the PDP-11 when PSIE is set.
b. That the DMA will cause an interrupt if IE is set when DMAREADY makes a low to high transition.
c. That Devices (TABLET, FUNCTION SW & LI, KEYBOARD) will cause interrupts if PSIE is set and the appropriate bit is set in the Device Interrupt Enable Register when the particular Device Request bit is set.

The diagnostic consists of five phases:

1. CLK/SYS/DEV INTERRUPT TEST
   Interrupt lines (*PSCLKINT, *PSSYSINT, *PSDEVINT) are jumpered to *PSDAT(15-13) on the PS2 backpanel. Each time one of the *PSDAT lines goes low, the associated interrupt jumpered to that line is generated and is transmitted to the PDP-11 for processing.
2. DMA INTERRUPT TEST
   Verifies that the DMA will cause an interrupt when DMAREADY makes a low to high transition when the IE bit is set.
3. Verifies that the Keyboard can cause a Device interrupt. Striking a key on the Keyboard should cause the interrupt. The interrupt routine reads the Keyboard data and outputs the Character to the console terminal.
4. Verifies that the Function Switches and Lights can cause a Device interrupt. A "switch setting change" should cause the interrupt. The interrupt routine reads the Switch Register and outputs the value to the console terminal and to the Light Register.

5. Verifies that the Tablet can cause a Device interrupt. The interrupt should occur each time the Tablet Request bit is set. The interrupt routine reads the Tablet status and data registers and outputs the values to the console terminal.

Program Description:

Phase 1: Verifies that PICTURE SYSTEM interrupts (CLOCK, SYSTEM and DEVICE) can be transmitted to and processed by the PDP-11. PICTURE SYSTEM memory location 0 is written throughout this phase to cause the *PSDAT (15-13) lines to transition causing the expected interrupts. Program steps are:

a. Ask for PS2 backpanel jumpers; wait for carriage return.
b. Set up the interrupt vectors and set PSIE and IE in the IOST register.
c. Load P.S. Memory location zero with 100000. Verify that only a single CLOCK interrupt occured.
d. Load P.S. Memory location zero with 40000. Verify that only a single SYSTEM interrupt occured.
e. Load P.S. Memory location zero with 20000. Verify that only a single DEVICE interrupt occured.
f. Load P.S. Memory location zero with 160000. Verify that a single CLOCK, SYSTEM and DEVICE interrupt occured.
g. Load P.S. Memory location zero with 100 random numbers. Verify that the correct interrupts occured.

f. Repeat steps (c) and (d) several times.

g. Output the following message:

DMA INTERRUPT TEST COMPLETE

RUNNING

h. Go to step (c).

Phase 3: Verifies that striking a key on the Keyboard will cause a Device interrupt when PSIE (IOST bit 8) is set and the appropriate bit is set in the Device Interrupt Enable Register. Program steps are:

a. Set up the interrupt vector and set PSIE in the IOST register. Steps (d) through (f) are in the interrupt handler.
b. Set the appropriate bit in the Device Interrupt Enable Register.
c. Wait for interrupt.
d. Read Keyboard data and output the character to the console terminal.
e. Clear the Keyboard Request bit.
f. Return from interrupt (go to step (c)).

Phase 4: Verifies that a "switch setting change" on the Function Switches and Lights will cause a Device interrupt when PSIE (IOST bit 8) is set and the appropriate bit is set in the Device Interrupt Enable Register. Program steps are:

a. Set up the interrupt vector and set PSIE in the IOST register. Steps (d) through (f) are in the interrupt handler.
b. Set the appropriate bit in the Device Interrupt Enable Register.
c. Wait for interrupt.
d. Read Switch Register and output the value to the console terminal.
e. Clear the FS&L Request bit.
f. Return from interrupt (go to step (c)).

h. If step (c), (d), (e), (f) or (g) fails, the following message is output.

```
1:  INTERRUPT ERROR;
A.  CLK INTERRUPTS EXP=XXX REC=XXXXXX
B.  SYS INTERRUPTS EXP=XXX REC=XXXXXX
C.  DEV INTERRUPTS EXP=XXX REC=XXXXXX
D.  DMA INTERRUPTS EXP=0 REC=XXXXXX
```

i. Go to step (c) 100 times.

j. Output the following message

```
CLK/SYS/DEV INTERRUPT TEST COMPLETE
RUNNING
```

l. Go to step (b).

Phase 2: Verifies that the DMA will interrupt the PDP-11 when IE (IOST bit 6) is set and DMAREADY (IOST bit 7) makes a low to high transition. Program steps are:

a. Set up the interrupt vectors and set PSIE and IE in the IOST register.

b. Set up to expect only a DMA interrupt.

c. Initiate a single DMA Active Output transfer (DMAWC=177777; DMABA=address of a PDP-11 Memory location containing zero; set GO in the IOST).

d. Verify that only a single DMA interrupt was received.

e. If step (d) fails, the following message is output.

```
1:  INTERRUPT ERROR;
A.  CLK INTERRUPTS EXP=0 REC=XXXXXX
B.  SYS INTERRUPTS EXP=0 REC=XXXXXX
C.  DEV INTERRUPTS EXP=0 REC=XXXXXX
D.  DMA INTERRUPTS EXP=1 REC=XXXXXX
```

Name: PS2 TABLET DIAGNOSTIC

Functional Description:

This diagnostic is used to verify basic tablet/controller and interface operation. Tablet status and data register values are output to the console terminal. The diagnostic consists of four phases:

1. TABLET VALUES TO TTY
   Tablet status and data register values are output to the console terminal each time the appropriate bit is set in the Device Request Register.
2. INHIBIT CONVERSIONS IF DATAREADY SET
   Verifies that the tablet data values do not change as long as DATAREADY (tablet status bit 3) is set.
3. INHIBIT CONVERSIONS IF INHIBIT SET
   Verifies that the tablet data values do not change as long as INHIBIT (tablet status bit 13) is set.
4. REMOTE TRIGGER WHEN PEN DOWN - TABLET VALUES TO TTY
   Verifies that tablet conversions take place only when TRIGGER (tablet status bit 12) is set if the tablet controller is in remote mode.

Program Description:

Phase 1: Allows the operator to verify basic tablet/control and interface operation by outputting tablet status and data register values to the console terminal each time the REQUEST bit is set. Program steps are:

   a. Wait for the REQUEST bit to set.
   b. Output tablet status and data register values to the console terminal.
   c. Clear the REQUEST bit.
   d. Go to step (a).

Phase 5: Verifies that a Device interrupt will occur if the Tablet Request bit gets set when PSIE (IOST bit 8) is set and the appropriate bit is set in the Device Interrupt Enable Register. Program steps are:

a. Set up the interrupt vector and set PSIE in the IOST register. Steps (d) through (f) are in the interrupt handler.
b. Set the appropriate bit in the Device Interrupt Enable Register.
c. Wait for interrupt.
d. Read Tablet data and output the values to the console terminal.

```
STAT=XXXXXX TABX=XXXXXX TABY=XXXXXX TABZ=XXXXXX
```

e. Clear the Tablet Request bit.
f. Return from interrupt (go to step (c)).

Error Analysis:

Error #	Prognosis
1:	This error indicates that an incorrect number of interrupts were processed by the PDP-11. If the error occured during phase 1, make sure that the correct number of interrupts are being received by the 195106-100 card and are being transmitted to the PDP-11. If interrupt transmission is correct, the problem is in the interrupt sense or interrupt control logic on the 195131-100 card. If the error occured during Phase 2, the problem is in the DMA interrupt sense logic on the 195131-100 card. No PICTURE SYSTEM interrupts should be generated or transmitted during phase 2 providing the jumpers from phase 1 have been removed from the PS2 backpanel.

with data received in step (d). If the data is not the same, the following message is output.

2: ERROR; TABLET VALUES CHANGED WHILE INHIBIT BIT SET;
EXPECTED: TABX=XXXXXX TABY=XXXXXX TABZ=XXXXXX
RECEIVED: TABX=XXXXXX TABY=XXXXXX TABZ=XXXXXX

h. Clear the INHIBIT bit.
i. Output "END PASS" message.
j. Go to step (b).

Phase 4: Verifies that Remote Trigger initiates tablet conversions. Program steps are:

a. Ask operator to put tablet controller in REMOTE mode. Wait for carriage return.
b. Issue a DTRESET to initialize tablet interface and controller.
c. Wait for PENSWITCH to be set (tablet status bit 4).
d. Verify that the REQUEST bit is not set.
e. If step (d) fails, the following message is output.

3: ERROR; TAB REQ BIT SET BEFORE REMOTE TRIGGER ISSUED.

f. Issue a Remote Trigger.
g. Wait for the REQUEST bit to set.
h. Output the tablet status and data register values to the console terminal.
i. Go to step (b).

Phase 2: Verifies that conversions are inhibited when DATAREADY is set. Program steps are:

a. Ask operator to put tablet controller in STREAM/*SWITCH mode. After receipt of carriage return, ask operator to move the pen to random positions on the tablet.
b. Wait for DATAREADY to be set (tablet status bit 3).
c. Read tablet data registers.
d. Timeout longer than 5 milliseconds.
e. Read tablet data registers. Compare the data with data received in step (c). If the data is not the same, the following message is output.

```
1:  ERROR; TABLET VALUES CHANGED WHILE DATAREADY SET;
EXPECTED: TABX=XXXXXX TABY=XXXXXX TABZ=XXXXXX
RECEIVED: TABX=XXXXXX TABY=XXXXXX TABZ=XXXXXX
```

f. Repeat steps (d) and (e) several times.
g. Output "END PASS" message.
h. Go to step (b).

Phase 3: Verifies that conversions are inhibited when the INHIBIT bit is set. Program steps are:

a. Ask operator to put tablet controller in STREAM/*SWITCH mode. After receipt of carriage return, ask operator to move the pen to random positions on the tablet.
b. Wait for DATAREADY to be set (tablet status bit 3).
c. Set the INHIBIT bit (tablet status bit 13).
d. Read the tablet data registers.
e. Clear the tablet request bit which clears DATAREADY.
f. Timeout longer than 5 milliseconds.
g. Read the tablet data registers. Compare the data

Identification: QSD103
Date: 12/27/76

Name: PS2 FUNCTION SW & LI DIAGNOSTIC

Functional Description:

This diagnostic is used to test the PS2 FUNCTION SWITCHES & LIGHTS option. It consists of three phases.

1. LIGHT REGISTER WRITE/READ/PSRESET
   Verifies that all bits of the Light Register can be set, cleared and reset.
2. SWITCH VALUE TO LIGHTS
   Verifies that a "switch setting change" will set the request bit. The Switch Register is read and loaded into the Light Register. Lights "ON" should correspond to switches "ON" after the first Light Register load.
3. SWITCH VALUE TO TTY
   Verifies that the switches are being "debounced". Each time the Request bit gets set, the Switch Register is read and the data received is output to the console terminal. Each switch should be able to be changed from "ON" to "OFF" to "ON" with only 2 values printed on the console terminal.

Program Description:

Phase 1: Verifies that all bits of the Light Register can be set, cleared and reset. Program steps are:

a. Load the Light Register with random data.
b. Read the Light Register and compare the data received with the data loaded in step (a).
c. If the data isn't the same, the following message is output.

1: LIGHT REG W/R ERROR; DATA EXP=XXXXXX DATA REC=XXXXXX

Error Analysis:

Error #	Prognosis
1:	This error indicates that eight tablet conversions were not inhibited when DATAREADY was set or that multiple reads of data register values produced different results. If tablet data is changing while the *INHIB signal (195181-100 card) is low then the problem is likely the tablet controller. If tablet data isn't changing then the problem is likely the data path from interface to PSBUS.
2:	This error indicates that either tablet conversions were not inhibited when INHIBIT was set or that the second read of data register values produced a different result than the first. If tablet data is changing while the *INHIB signal (195181-100 card) is low then the problem is likely the tablet controller. If tablet data isn't changing then the problem is likely the data path from interface to PSBUS.
3:	This error indicates that a tablet conversion took place before a REMOTE TRIGGER pulse was sent to the tablet controller. The problem is likely in the tablet controller .

Error Analysis:

Error #	Prognosis
1.	This error indicates that one or more bits of the Light Register (located on the 195191-100 card) could not be either written or read correctly. The problem could be the data path to the register, register control logic, the register itself, or the data path from the register.
2.	This error indicates that the Light Register (located on the 195191-100 card) could not be reset to zero by PSRESET. Check the register reset logic.

d. Repeat steps (a) and (b) several times.
e. Load the Light Register with 177777.
f. Do a PSRESET and read the Light Register.
g. If the register did not contain zero, the following message is output.

2: LIGHT REG PSRESET ERROR; DATA EXP=0 DATA REC=XXXXXX

h. Repeat steps (e) and (f) several times.
i. Stop - phase complete.

Phase 2: Verifies that the Request bit gets set when a "switch setting change" occurs and that the Switch Register can be read correctly. Program steps are:

a. Wait for the Request bit to set.
b. Read the Switch Register and load the data received into the Light Register.
c. Clear the Request bit.
d. Go to step (a).

Phase 3: Verifies that the switches are being "debounced". Program steps are:

a. Wait for the Request bit to set.
b. Read the Switch Register and output the data received to the console terminal.
c. Clear the Request bit.
d. Go to step (a).

e. Clear the Request bit.
f. Go to step (b) unless the last character was a "space".
g. Go to step (a) if not finished verifying all three types of characters.
h. Output the following message.

CHARACTER VERIFICATION COMPLETE

i. Go to step (a).

Phase 2: Outputs the Keyboard code to the console terminal each time the Keyboard Request bit gets set. Program steps are:

a. Wait for the Request bit to set.
b. Read the Keyboard data register and output the value to the console terminal.
c. Clear the Request bit.
d. Go to step (a).

Phase 3: Outputs the character to the console terminal each time the Keyboard Request bit gets set. Program steps are:

a. Wait for the Request bit to set.
b. Read the Keyboard data register and output the character to the console terminal.
c. Clear the Request bit.
d. Go to step (a).

Error Analysis:

Error #	Prognosis
1:	This error indicates that the data received from the Keyboard/interface did not agree with the data in the table of expected Keyboard codes. If the data coming from the Keyboard is correct then the problem is in the data path from interface to PSBUS.

Name: PS2 KEYBOARD DIAGNOSTIC

Functional Description:

This diagnostic is used to test the PS2 KEYBOARD option. It consists of 3 phases:

1. CHARACTER VERIFICATION
   Verifies that all Character Codes (lower case, upper case and control character) received from the Keyboard are correct by comparing them to appropriate tables of expected codes. Each table expects keys to be depressed in a left to right fashion beginning with the "ESC" key. Refer to Table QSD104-1 and Figure QSD104-1.
2. KEYBOARD OCTAL CHARACTER CODE TO TTY
   Outputs the octal Character Code to the console terminal each time the keyboard Request bit gets set.
3. KEYBOARD CHARACTER TO TTY
   Outputs the Keyboard Character to the console terminal each time the Keyboard Request bit gets set.

Program Description:

Phase 1: Verifies that all Character Codes received from the Keyboard are correct. Program steps are:

a. Ask the operator to start Character verification (lower case, upper case or control characters).
b. Wait for the Request bit to set.
c. Read the Keyboard data register and compare the data received with data in the appropriate character code table.
d. If the data is not correct, the following message is output.

1: ERROR; DATA EXP=XXX DATA REC=XXX

TABLE QSD104-1 (cont.)

	UNSHIFTED		SHIFTED		CONTROL	
KEY NO.	CHAR.	OCTAL CODE	CHAR.	OCTAL CODE	CHAR.	OCTAL CODE
47	f	146	F	106	ACK	006
48	g	147	G	107	BEL	007
49	h	150	H	110	BS	010
50	j	152	J	112	LF	012
51	k	153	K	113	VT	013
52	l	154	L	114	FF	014
53	;	073	+	053		
54	:	072	*	052		
55	]	135	}	175	GS	035
56	LF	012	LF	012		
57	DEL	177	DEL	177		
62	CTRL (Mode Selection Key)					
64	SHIFT (Mode Selection Key)					
65	z	172	Z	132	SUB	032
66	x	170	X	130	CAN	030
67	c	143	C	103	ETX	003
68	v	166	V	126	SYN	026
69	b	142	B	102	STX	002
70	n	156	N	116	SO	016
71	m	155	M	115	CR	015
72	,	054	<	074		
73	.	056	>	076		
74	/	057	?	077		
76	SHIFT (Mode Selection Key)					
77	CTRL (Mode Selection Key)					
81	SP	040	SP	040	SP	040

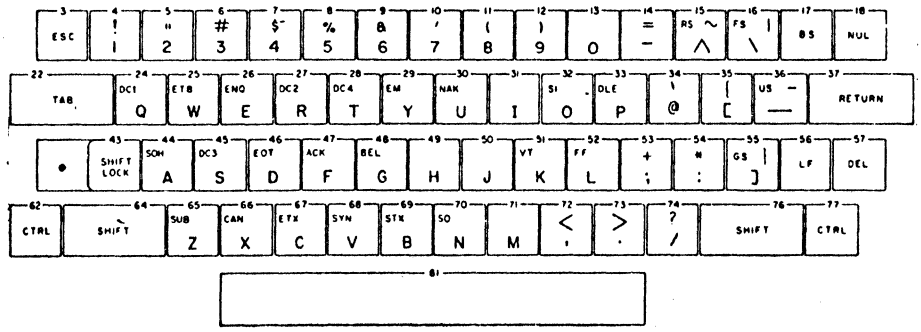


Figure OSD104-1

	UNSHIFTED		SHIFTED		CONTROL	
KEY NO.	CHAR.	OCTAL CODE	CHAR.	OCTAL CODE	CHAR.	OCTAL CODE
3	ESC	033	ESC	033		
4	1	061	!	041		
5	2	062	"	042		
6	3	063	#	043		
7	4	064	$	044		
8	5	065	%	045		
9	6	066	&	046		
10	7	067	'	047		
11	8	070	(	050		
12	9	071	)	051		
13	0	060				
14	-	055	=	075		
15	^	136	~	176	RS	036
16	\	134	|	174	FS	034
17	BS	010	BS	010		
18	NUL	000	NUL	000		
22	HT	011	HT	011		
24	q	161	Q	121	DC1	021
25	w	167	W	127	ETB	027
26	e	145	E	105	ENQ	005
27	r	162	R	122	DC2	022
28	t	164	T	124	DC4	024
29	y	171	Y	131	EM	031
30	u	165	U	125	NAK	025
31	i	151	I	111	HT	011
32	o	157	O	117	SI	017
33	p	160	P	120	DLE	020
34	@	100	`	140	NUL	000
35	[	133	{	173	ESC	033
36	_	137	_	137	US	037
37	CR	015	CR	015		
43	SHIFT LOCK - (Mode Selection Key)					
44	a	141	A	101	SOH	001
45	s	163	S	123	DC3	023
46	d	144	D	104	EOT	004

check the function of the analog device generating the channel data. No error checking is done. The phase runs indefinitely. The program steps are:

a. Get number of channels to display
b. Get new channel data
c. Convert data to ASCII
d. Draw channel data on monitor
e. Repeat steps (b), (c), and (d) indefinitely

Phase 2: This phase prints out 16 channels of received data. No error checking is done. This phase must be requested. The program steps are:

a. Get new channel data
b. Print channel data
c. Repeat step (b) 16 times
d. End of phase

Phase 3: This phase lets the operator calibrate the OFFSET potentiometer to a null level so the Data Acquisition Module can generate channel data correctly. The operator is guided through the procedure by the program. No error checking is done. This phase must be requested. The program steps are:

a. Give operator instructions
b. Get count of times to print data
c. Get new channel data
d. Print CHAN 3 and 4 data
e. Repeat steps (c), (d) until count runs out
f. End of phase

Phase 4: This phase lets the operator calibrate the GAIN potentiometer to a specific level so the Data Acquisition Module can generate channel data across the full

Date: 5/26/77

Name: UNIVERSAL ANALOG INTERFACE DIAGNOSTIC

Functional Description:

This diagnostic is used to test and calibrate the Universal Analog Interface card (195201-100) and for displaying channel data on the monitor to check the function of any analog device driving the interface. The program will ask the operator to respond with certain parameters as different phases are requested and executed. The diagnostic consists of five phases:

1. Visual display of channels
2. Print out of channels
3. Offset calibration
4. Gain calibration
5. Expected/received data comparison check

Set Up Procedure:

Phases 1 and 2 above are usually used with the system set up for normal operation of the analog device(s).

Phases 3,4 and 5 require the following modification:

> Prepare the 195201-100 card for testing by removing cables from J1 and/or J2. Install a 10 volt resistor ladder (195204-100) into J2. Set the dip switches for pseudo - differential input mode and +-10 volt operation.

Program Description:

Phase 1: This phase expects a working Universal Analog Interface card to be used. The desired number of channels are displayed on the monitor so the operator can visually

Identification: QSD106
Date: 3/30/77

Name: PS2 RRD KEYBOARD DIAGNOSTIC

Functional Description:

This diagnostic is used to verify PS2 RRD KEYBOARD operation. It consists of 2 phases.

1. CHARACTER VERIFICATION
   Verifies that all Character Codes (lower case, upper case, control and auxiliary) received from the Main and Auxiliary Keyboards are correct by comparing them to appropriate tables of expected codes. Each table expects keys to be depressed in a left to right fashion (in numeric order) beginning with the upper left hand key. Refer to Figure QSD106-1.
2. KEYBOARD OCTAL CHARACTER CODE TO TTY.
   Outputs the octal Character Code to the console terminal each time the Keyboard Request bit gets set.

Program Description:

Phase 1: Verifies that all Character Codes received from the Main and Auxiliary Keyboards are correct. Program steps are:

a. Ask the operator to start Character verification (lower case, upper case, control or auxillary characters).
b. Wait for the Request bit to set.
c. Read the Keyboard data register and compare the data received with data in the appropriate character code table.
d. If the data is not correct, the following message is output.

1: ERROR; DATA EXP=XXXXXX DATA REC=XXXXXX

e. Clear the request bit.

voltage range. The operator is guided in the procedure by the program. No error checking is done. This phase must be requested. The program steps are:

a. Give operator instructions and/or get count of times to print data
b. Get new channel data
c. Print CHAN 0 data
d. Repeat steps (b), (c) until count runs out
e. End of phase

Phase 5: This phase verifies that the Data Acquisition Module is receiving and converting specific analog voltages to digital values correctly. Error checking is used. To loop on error type "-1" as the response to "WHICH MESSAGES?" This phase must be requested. The program steps are:

a. Give operator instructions
b. Get channel data
c. Compare received data with expected (+ or - a specified tolerance)
d. If the data is out of tolerance the following message is printed:

```
CHAN XX EXPT (+-XX)=XXXXX RECD=XXXXX DIFF=XXXXX
```

e. Repeat steps (c), (d) 16 times to check all channels
f. End of phase/end of program

Prognosis:

Phase 5 is a go/no go test. The card is simple with the bulk of the logic contained in the Data Acquisition Module. A problem can not usually be isolated to the bug level. Inspection of problem data for different channels can help the operator determine whether certain bits are in error; if several bits are stuck high or low it may indicate a failing RAM.

Error Analysis:

Error #	Prognosis
1:	This error indicates that the data received from the Keyboard/interface did not agree with the data in the table of expected Keyboard codes. If the data coming from the Keyboard is correct then the problem is in the data path from the Keyboard interface to PSBUS.

f. Go to step (b) unless last character in the table.
g. Go to step (a) if not finished verifying all four types of characters.
h. Output the following message:

CHARACTER VERIFICATION COMPLETE

i. Go to step (a).

Phase 2: Outputs the Character Code to the console terminal each time the keyboard request bit gets set. Program steps are:

a. Wait for the Request bit to set.
b. Read the Keyboard data register and output the value to the console terminal.
c. Clear the Request bit.
d. Go to step (a).

Name: PS2 32-BIT FUNCTION SW & LI DIAGNOSTIC

Functional Description:

This diagnostic is used to test the PS2 32-BIT FUNCTION SWITCHES & LIGHTS option. It consists of three phases.

1. SWITCH VALUE TO LIGHTS
   Verifies that a "switch setting change" will set the request bit. The Switch Register is read and loaded into the Light Register. Lights "ON" should correspond to switches "ON" after the first Light Register load.
2. SWITCH VALUE TO TTY
   Verifies that the switches are being "debounced". Each time the Request bit gets set, the Switch Register is read and the data received is output to the console terminal. Each switch should be able to be changed from "ON" to "OFF" to "ON" with only 2 sets of values printed on the console terminal.
3. LIGHT REGISTER WRITE/READ/PSRESET
   Verifies that all bits of the Light Register can be set, cleared and reset.

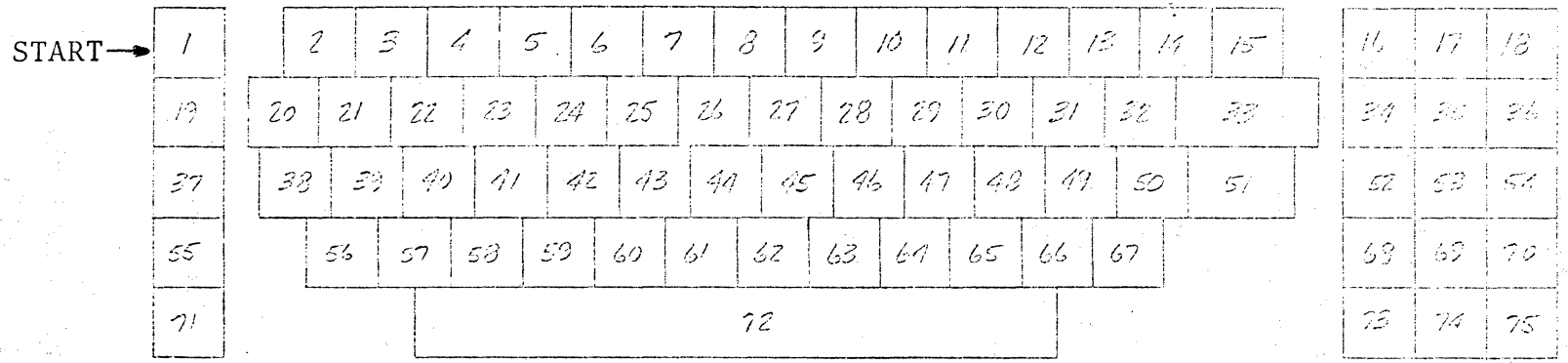
Program Description:

Phase 1: Verifies that the Request bit gets set when a "switch setting change" occurs and that the Switch Register can be read correctly. Program steps are:

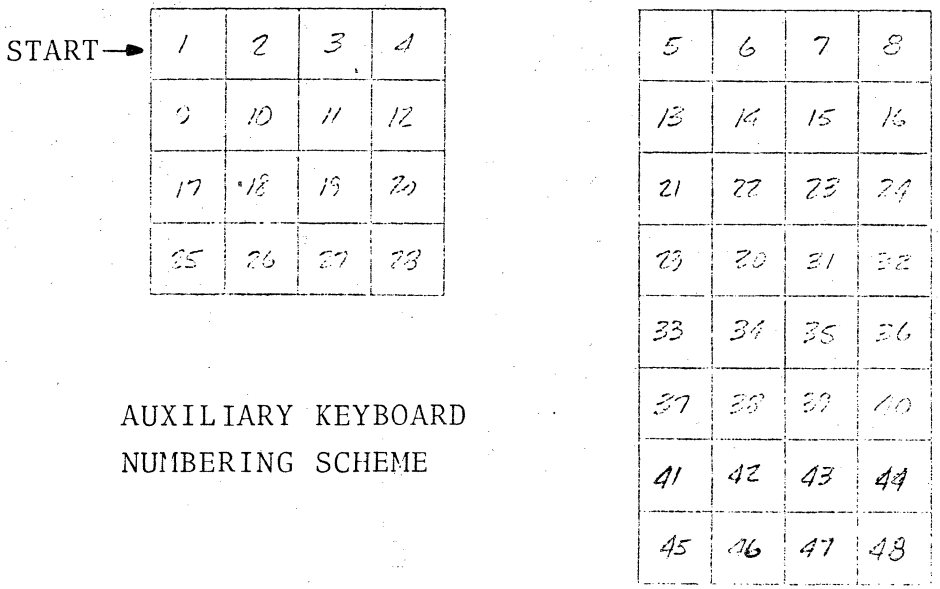
- a. Wait for the Request bit to set.
- b. Read the Switch Register and load the data received into the Light Register.
- c. Clear the Request bit.
- d. Go to step (a).

Phase 2: Verifies that the switches are being "debounced". Program steps are:

- a. Wait for the Request bit to set.
- b. Read the Switch Register and output the data received to the console terminal.



MAIN KEYBOARD NUMBERING SCHEME



AUXILIARY KEYBOARD
NUMBERING SCHEME

Figure QSD106-1

Name: PS2 LIGHT PEN DIAGNOSTIC

Functional Description:

Successful execution of this diagnostic indicates that the PS2 LIGHT PEN is operational. It verifies the following:

a. Registers on the PSBUS operate correctly.
b. Light Pen can detect light.
c. Proper interaction between the line generator and the Light Pen.

NOTE: This diagnostic should not be run until the correct operation of the PSBUS and the line generator is verified. Phases 2, 3, 10, 11, 12, 13, and 14 do not terminate.

This diagnostic consists of fourteen phases.

1. Interrupt enable bit test.
   Verifies that the interrupt enable bit can be set, cleared and reset.
2. Interrupt request bit test.
   Verifies that the interrupt request bit can be set by the tipswitch and cleared by a PSBUS access.
3. Control and status register test (tipswitch bit).
   Verifies that the tipswitch bit in the control and status register reflects the state of the tipswitch.
4. Segment Name Capture.
   Verifies that the Light Pen hardware can properly capture segment names.
5. Endpoint catcher test.
   Verifies that the Light Pen hardware can properly capture the endpoints that are sent through the line generator.
6. Character counter test.
   Verifies that the Light Pen hardware can properly count the number of characters sent through the line generator.

c. Clear the Request bit.
d. Go to step (a).

Phase 3: Verifies that all bits of the Light Register can be set, cleared and reset. Program steps are:

a. Ask the operator to open dipswitch in Position 10.1 (195261).
b. Load the Light Register with random data.
c. Read the Light Register and compare the data received with the data loaded in step (a).
d. If the data isn't the same, the following message is output.

1: LIGHT REG W/R ERROR; DATA EXP=XXXXX DATA REC=XXXXX

e. Repeat steps (a) and (b) several times.
f. Load the Light Register with 177777.
g. Do a PSRESET and read the Light Register.
h. If the register did not contain zero, the following message is output.

2: LIGHT REG PSRESET ERR; DATA EXP=0 DATA REC=XXXXX

i. Repeat steps (f) and (g) several times.
j. Stop - phase complete.
k. Ask the operator to close dipswitch 10.1 (195261)

Phase 2: Verifies that the interrupt request bit can be set by the tipswitch and cleared by a PSBUS access. Program steps are:

a. The program outputs the following message:

TOGGLE THE LIGHT PEN TIPSWITCH

b. The program now waits for the interrupt request bit to set.
c. When the user closes the tipswitch, the interrupt request bit will set.
d. The request bit is cleared and the following message is output:

INTERRUPT REQUEST BIT SET

e. When the user releases the tipswitch, the interrupt request bit will set and (d) will occur again.
f. Program continues at step (b).

NOTE: This phase does not terminate.

Phase 3: Verifies that the tipswitch bit in the control and status register reflects the state of the tipswitch. Program steps are:

a. Output the following message:

TOGGLE THE LIGHT PEN TIPSWITCH

b. Wait until the interrupt request bit is set.
c. When the interrupt request bit is set, read the light pen control and status register.
d. Display the light pen control and status register on the screen.
e. Go to (b).

NOTE: This phase does not terminate.

7. Control and status register tests (character bit).
   Verifies that when the line generator is processing characters, the character bit in the control and status register will be set.
10. Light Pen hit test.
    Verifies that the Light Pen can detect lines drawn on the screen.
11. Line length counter test.
    Verifies that the line length counter gets approximately the right number in it when the Light Pen detects a line.
12. Line length subcount test.
    Verifies that the line length subcounter gets the right number in it when the Light Pen detects a line.
13. Segment name (line) integrity.
    Verifies that lines can be uniquely labeled by refresh controller segment names.
14. Segment name (character) integrity.
    Verifies that characters can be uniquely labeled by refresh controller segment names.

Program Description:

Phase 1: Verifies that the interrupt enable bit can be set, cleared and reset. Program steps are:

a. Randomly write a 1 or Ø into the interrupt enable bit.
b. Check that the bit was written properly.
c. If the bit was not written properly, the following message is output:

1: INTERRUPT ENABLE BIT ERROR; DATA EXP=XXXXXX DATA REC=XXXXXX

d. Do (a) and (b) several times.
e. Stop - Phase complete.

Phase 5: Verifies that the light pen hardware can properly capture the end points that are sent through the line generator. Program steps are:

a. A segment name that causes the line generator to reestablish its position after every move is sent to the line generator.
b. The commands MOVETO (1,2) and DRAW TO (3,4), are sent to the line generator.
c. The light pen hardware is read to see if the endpoints were captured properly.
d. If they are not captured properly the following message is output:

4: END POINT CATCHER STATE A ERROR; DATA EXP=XXXXX,XXXXX,XXXXX,XXXXX
DATA REC=XXXXX,XXXXX,XXXXX,XXXXX

e. The command DRAWTO (5,6) is sent to the line generator.
f. The light pen hardware is read to see if the endpoints were captured properly.
g. If they were not captured properly, the following message is output:

5: ENDPOINT CATCHER STATE B ERROR; DATA EXP=XXXXX,XXXXX,XXXXX,XXXXX
DATA REC=XXXXX,XXXXX,XXXXX,XXXXX

h. The command DRAWTO (7,10) is sent to the line generator.
i. The light pen hardware is read to see if the endpoints were captured properly.
j. If they were not captured properly, the following message is output:

6: ENDPOINT CATCHER STATE C ERROR; DATA EXP=XXXXX,XXXXX,XXXXX,XXXXX
DATA REC=XXXXX,XXXXX,XXXXX,XXXXX

k. The command DRAWTO (X,Y) where X and Y are random numbers, is sent to the line generator.

Phase 4: Verifies that the light pen hardware can properly capture segment names. Program steps are:

a. Make a refresh controller command with a random segment name.
b. Send the command to the line generator passive input port.
c. Read the segment name register in the light pen hardware and compare it to the one sent to the line generator.
d. If the names are different, the following message is output:

2: SEGMENT NAME DATA ERROR: DATA EXP=XXXXXX DATA REC=XXXXXX

e. Steps (a), (b) and (c) are repeated several times.
f. A segment name is generated and sent to the line generator.
g. Several other segments are sent, but these segments have the segment name take bit turned off so they do not disturb the segment sent to the line generator in step (f).
h. Read the segment name register in the light pen hardware and compare it to the one sent out in (f).
i. If the names are different, the following message is output:

2: SEGMENT NAME TAKE BIT ERROR; DATA EXP=XXXXXX DATA REC=XXXXXX

j. Do steps (e), (f), (g) and (h) several times, then the following message is output.

SEGMENT NAME VERIFICATION COMPLETE

k. Stop - Phase complete.

11: CHARACTER COUNTER OVERFLOW ERROR; DATA EXP=XXXXX DATA REC=XXXXX

h. Do steps (e) and (f) several times.
i. The following message is output:

CHARACTER COUNTER TEST COMPLETE

j. Stop - Phase complete.

Phase 7: Verifies that when the line generator is processing characters, the character generator bit in the control and status register will be set. Program steps are:

a. A segment name is sent to the line generator to clear the character counter and character hit bit.
b. Several characters are sent to the line generator.
c. The character bit in the control and status register is checked to see if it is set.
d. If it is not set, the following message is output:

12: CHARACTER BIT ERROR; DATA EXP=XXXXX DATA REC=XXXXX

e. Steps (a), (b) and (c) are executed several times.
f. The following message is output:

CONTROL AND STATUS REGISTER (CHARACTER BIT) TEST COMPLETE

g. Stop - Phase complete.

Phase 10: Verifies that the light pen can detect lines drawn on the screen. Program steps are:

a. The following message is output:

POINT THE LIGHT PEN AT THE LINE

l. The light pen hardware is examined to see if the endpoints were captured properly.

m. If they were not captured properly, the following message is output:

7: ENDPOINT CATCHER DATA ERROR; DATA EXP=XXXXX,XXXXX,XXXXX,XXXXX DATA REC=XXXXX,XXXXX,XXXXX,XXXXX,

n. Do steps (k) and (l) several times.

o. Output the following message:

BACKPANEL PIN 50.60 SHOULD BE BETWEEN 3-5 VOLTS

p. The operator should verify that the backpanel pin is between three and five volts.

q. The program waits for a carriage return.

r. Stop - Phase complete.

Phase 6: Verifies that the light pen hardware can properly count the number of characters sent through the line generator. Program steps are:

a. Send a random number of characters to the line generator.

b. Read the light pen hardware and verify that it reports the proper number of characters.

c. If there is a difference, output the following message:

10: CHARACTER COUNTER ERROR; DATA EXP=XXXXX DATA REC=XXXXX

d. Do steps (a) and (b) several times.

e. A number of characters large enough to overflow the character counter is sent to the line generator.

f. The character over flow bit is verified.

g. If it is not set, the following message is output:

b. The line length subcount and count registers are read and displayed on the screen in octal. The subcount will appear nearest to the top of the screen and the count will appear nearest to the bottom of the screen.
c. Two lines are drawn on the screen.
d. The interrupt request bit is polled until it is set indicating a light pen hit has occurred.
e. The program returns to step (b).
NOTE: The values for the count register are the same as in Phase 11. The subcount should be zero when the light pen is pointed at the left end of the lines and should be equal to one minus the number in the count register when the light pen is pointed at the right end of the line. This can be adjusted by trimming the variable delay line on the 195271-100 card. See the 195271-100 card description for further details. This phase does not exit.

Phase 13: Verifies that lines can be uniquely labeled by refresh controller segment names. Program steps are:

a. The following message is output:

POINT THE LIGHT PEN AT THE LINE

b. Twelve short lines and four dots are displayed on the screen.
c. The interrupt request bit is polled until it is set indicating a light pen hit has occurred.
d. The segment name register in the light pen hardware is read and the segment name is retrieved.
e. The segment name is used to identify which of the twelve lines or four dots caused the hit. The line or dot that caused the hit is then redrawn so that it will become brighter and can be identified.

b. The light pen control and status register is read.
c. The octal value of the control and status register is displayed on the screen.
d. A line is displayed on the screen.
e. When the user points the pen at the line, the corresponding bit should appear in the octal value displayed on the screen.
f. The program goes back to step (b).
NOTE: This phase does not terminate.

Phase 11: Verifies that the line length counter gets approximately the right number in it when the light pen detects a line. Program steps are:

a. The following message is output:

POINT THE PEN AT THE LINE

b. The line length register is read and displayed in octal on the screen.
c. Two lines are drawn on the screen.
d. The interrupt request bit is polled until it is set indicating a light pen hit has occurred.
e. The program returns to step (b).
NOTE: Two different numbers should appear on the screen, one equal to 171, the other is equal to 362. These numbers represent the lengths of the lines on the screen and may vary by ±2 units. The phase does not terminate.

Phase 12: Verifies that the line length subcounter gets the right number in it when the light pen detects a line. Program steps are:

a. The following message is output:

POINT THE LIGHT PEN AT THE LINE

Error Analysis:

Error #	Prognosis
1:	This error indicates that the interrupt enable bit is not reading or writing properly. Check the dipswitch settings on the 195273-100 card.
2:	This error indicates that the segment names are not returned as they were sent out. Check the segment name register on the 195271-100 card. The 195272 cable and the multiplexers on the 195273-100 card which connects to the PSBUS.
3:	This error indicates that the segment name take bit (on the 195271-100 card) is not operating correctly.
4:, 5:, 6:	These errors indicate that the endpoint catcher state machine is not operating correctly (on the 195271-100). Check its inputs and the 195272 cable.
7:	This error indicates that the endpoints cannot be properly retrieved by the PSBUS. Check the memories on the 195271-100 card.
10:	This error indicates that the character counter does not count characters properly. Check the character counter and pipeline on the 195271 card and also check the 195272 cable.
11:	This error indicates that the character counter overflow bit is not being set by the overflow condition. This bit is on the 195271-100 card.

f. The program returns to step (b)
NOTE: This phase does not exit.

Phase 14: Verifies that characters can be uniquely labeled by refresh controller segment names. Program steps are:

a. The following message is output:

POINT THE PEN AT A CHARACTER

b. Sixteen characters are displayed on the screen.
c. The interrupt request bit is polled until it is set indicating a light pen hit has occurred.
d. The segment name register in the light pen hardware is read and the segment name is retrieved.
e. The segment name is used to identify which of the sixteen characters caused the hit. The character that caused the hit is then redrawn so that it will be brighter than the other characters and can be identified.
f. The program returns to step (b).
NOTE: This phase does not exit.

NAME: WRITEABLE CONTROL STORE DIAGNOSTIC

Functional Description:

This diagnostic is used to test the WRITEABLE CONTROL STORE cards (195124). It also loads the CONTROL STORE with the micro-code for standard operation (this allows the remaining diagnostics to function properly).

NOTE: Phases 1, 2 and 3 of QSD004 must run successfully before this test will operate properly.

The diagnostic consists of four phases:

1. Control Store Address Test
2. Control Store Checker Board
3. Control Store Swapped Checkerboard Test
4. Control Store Standard Micro-Code Loading & Testing

Program Description:

Phase 1: This phase loads each 16-bit word of each 96-bit control store word with its corresponding RAM address. The address is written into both bytes of each 16-bit word so both RAM cards can be checked as a unit. The data is read back and appears in each byte. The program steps are:

a. Clear the address count.
b. Write the address count value into both bytes of each of the six 16-bit words at the RAM address pointed to by the address count.
c. Increment the address count.
d. Repeat steps b and c 256 times.
e. Clear the address count.
f. Read the 96-bit RAM word pointed to by the address count.

Error #	Prognosis
* 12:	This error indicates that the character bit in the control and status register is not operating properly. It should set when characters are being processed by the line generator and cleared when they are not. Check the 195271-100 card, the 195272 cable and the 195273-100 card.

with the expected data. The program steps are:

a. Write the contents of CROM into the CONTROL STORE.
b. Clear the address count.
c. Read the 96-bit RAM word pointed to by the address count.
d. Compare each of the six 16-bit words with the expected "CROM" value.
e. If an error occurs in any word, the following message is printed:

4: RAM CODE ERR; WORD=X @ ADDR=XXX EXPT=XXXXXX RECD=XXXXXX

f. Increment address count.
g. Repeat steps c, d, e and f 256 times.
h. End of phase 4 - End of program.

g. Compare each byte of each of the six 16-bit words with the address count.
h. If an error occurs in any word, the following message is printed:

1: RAM ADDR ERR; WORD=X @ ADDR=XXX EXPT=XXXXXX RECD=XXXXXX

i. Increment address count.
j. Repeat f, g, h and i 256 times.
k. End of phase.

Phase 2: This phase loads each of the six 16-bit words of each control store word with a checkerboard pattern at all 256 addresses. The data is read back and checked to see if the checkerboard pattern appears in each of the six 16-bit words at each address. The program steps are the same as Phase 1 with two exceptions:

1. A checkerboard pattern is used as the write/compare data in steps b and g.
2. The error message printed in step h is:

2: RAM CHK/BOARD ERR; WORD=X @ ADDR=XXX EXPT=XXXXXX RECD=XXXXXX

Phase 3: This phase loads each of the six 16-bit words of each control store word with a swapped checkerboard pattern at all 256 addresses. The data is read back and checked with expected data. The program steps are the same as Phase 1 steps with two exceptions.

1. A swapped checkerboard pattern is used as the write/compare data in steps b and g.
2. The error message printed in step h is:

3: RAM SWAP/CHKBOARD ERR; WORD=X @ ADDR=XXX EXPT=XXXXXX RECD=XXXXXX

Phase 4: This phase loads each of the six 16-bit words of each control store word with the standard micro-code for normal opeations. The data is read back and checked

Identification: QSD110
Date: 9/8/77

Name: PS2 RTI Diagnostic

MUST ASSIGN RTO } RSX 11
> ASN TTX: = RTO: }

MH3 = 777560 } RT11
MH4 = 0 }

Functional Description:

This diagnostic is used to test the PS2 RTI (Remote Terminal Interface). It consists of 8 phases.

1. RTI Output
   This phase sends 100 lines of characters starting with octal 40 and going through octal 160. Characters on the right hand side of the screen will be clipped.
2. Keyboard Verification
   Verifies that all Character Codes (lower case, upper case and control character) received from the Keyboard are correct by comparing them to appropriate tables of expected codes. Each table expects keys to be depressed in a left to right fashion beginning with the "ESC" key. Refer to Table QSD104-1 and Figure QSD104-1.
3. ECHO
   This phase waits for characters from the keyboard and returns all characters that are sent to it. This can also be used to test the clear and remote/local.
4. DC1 Visual
   This phase sends a DC1 character (octal=21) to fill the RTI buffer with line generator commands. This subroutine puts up a pattern of concentric boxes with the characters "RTI" displayed in the center box.
5. RTI Active to RTI Passive Port (Serial Interface Option only)
   This phase writes data into the RTI Passive Input Port using the RTI Active Output Port.
6. DIO RTI Passive Port (Serial Interface Option only)
   This phase writes data into the RTI Passive Input Port using the DIO Active Output Port.

Error Analysis:

Error #	Prognosis
1:	This error indicates that the RAM cannot be addressed correctly, written and read correctly or that the RAM has some bad bits.
2:	If error 1: has not indicated an addressing or write/read error, then this error will usually indicate the RAM has some bad bits.
3:	This error will usually verify that error 2: was a valid error.
4:	This error indicates the RAM cannot be written and read correctly with the appropriate micro-code.

NOTE: The error message printed "WORD=X" has the following meanings:

WORD=0	DOIT bits 95-80-
WORD=1	DOIT bits 79-64
WORD=2	DOIT bits 63-48
WORD=3	DOIT bits 47-32
WORD=4	DOIT bits 31-16
WORD=5	DOIT bits 15-0

f. Go to step (a) if not finished verifying all three types of characters.
g. Output the following message.

CHARACTER VERIFICATION COMPLETE

h. Go to step (a).

Phase 3: Echos the characters that are typed.
a. Do steps b and c 100 times, then end Phase 3.
b. Wait for a character.
c. Send that character back to the RTI.

Phase 4: Puts up 31 concentric boxes with the RTI! displayed in the middle.
a. Send a DC1 (octal=21) character.
b. Send a count character (octal=0, for 256 words).
c. Send 512 characters that describe 16 boxes.
d. Send a DC1 (octal=21) character.
e. Send a count character (octal=12, for 246 words).
f. Send 480 characters that describe 15 boxes.
g. Send 12 characters which move to the center of the screen, write the characters RTI!, then move the cursor back to the center of the screen.
h. End Phase 4.

Phase 5: Verifies that the RTI Active Port can write the RTI Passive Port.
a. Do steps (b-h) 100 times, then end Phase 5.
b. Send a DC2 character (octal=22) to the RTI.
c. Send a count character (octal=377, for 1 word).
d. Send two characters which represent the address of the RTI Pass-Port.
e. Send two characters with random data in them.
f. Wait for the RTI to receive two characters.
g. Compare the characters that were sent with the ones that were received. If they are equal, go to step (b).

7. RTI Active Port to DMA Pass-Port (Serial Interface Option only)
   This phase uses the RTI Active Output Port to write data into the DMA Passive Input Port. The DMA transfers this data into the host computer memory for verification.
8. RTI Active Input & Reset (Serial Interface Option only)
   This phase writes the DMAPSA with the RTI Active Output Port and verifies the DMAPSA with the DIO. The DIO writes the DMAPSA and the RTI Active Input Port reads it. The DIO writes the DMAPSA, the RTI sends a reset command and the DIO verifies that the reset occurred.

Program Description:

Phase 1: Sends characters to the RTI for display.

a. Sound the bell (octal 7) character.
b. Do steps c and d 100 times, then end Phase 1.
c. Send the characters 40 octal thru 160 octal.
d. Send carriage return and line feed.

Phase 2: Verifies that all Character Codes received from the Keyboard are correct. Program steps are:

a. Ask the operator to start Character verification (lower case, upper case or control characters).
b. Wait for a character to arrive.
c. Read the Keyboard data register and compare the data received with data in the appropriate character code table.
d. If the data is not correct, the following message is output.

1: KEYBOARD DATA; DATA EXP=XXX DATA REC=XXX

e. Go to step (b) unless the last character was a "space".

g. Wait until the DMA has completed the transfer.
h. Compare the data in the host buffer with the characters sent and go to step (b) if equal, otherwise, go to step (i).
i. Output the following message:

4: RTI TO DMA PASS-PORT ERRPR: DATA EXP=XXX DATA REC=XXX

j. Then end Phase 7.

Phase 8: Writes the DMAPSA, reads the DMAPSA and sends a PS RESET.

a. Do steps (b-v) 100 times, then end Phase 8.
b. Get a random number and write its complement in the DMAPSA using the DIO.
c. Send a DC2 character (octal=22) to initiate an Active Output transfer.
d. Send a count character (octal=377, for a 1 word transfer).
e. Send two characters which represent the PSBUS address of the DMAPSA.
f. Send two characters which represent the random number from step (b).
g. Read the DMAPSA via DIO and see if the data written into it in step (b) is the same. If it is different, go to step (j), otherwise go to step h.
h. Increment a time out counter and check if it has reached a time out value. If it hasn't, go to step (g), otherwise, output the following message:

5: RTI DATA SEND ERROR; DATA EXP=XXX DATA REC=XXX

i. Then continue to step ( ).
j. Go to step (1).

h. Since the characters were found to be not equal in step (g), output the following message:

2: RTI ACTIVE TO RTI PASSIVE ERROR; DATA EXP=XXX DATA REC=XXX

i. Then end Phase 5.

Phase 6: Send data from the host computer interface to the RTI Passive Input Port.

a. Do steps (b-e) 100 times, then end Phase 6.
b. Write a 16 bit random number into the RTI Pass-Port via the host computer DIO interface.
c. Wait for two characters to come from the RTI.
d. Compare these two characters with the data sent in step (b), go to step (b) if they are equal otherwise, go to step (e).
e. Output the following message:

3: DIO TO RTI ERROR; DATA EXP=XXX DATA REC=XXX

f. Then end Phase 6.

Phase 7: Send data from the RTI back to the host computer via the DMA Passive Input Port.

a. Do steps (b-i) 100 times then end Phase 7.
b. Set up a 1 word DMA transfer to transfer a word from the DMA Passport to a 1 word buffer in the host computer.
c. Send a DC2 character (octal=22) to initiate an active output transfer.
d. Send a count character (octal=377, for 1 word).
e. Send a two character string which represents the PSBUS address of the DMAPIP.
f. Send two characters which represent a random 16 bit number.

v. Check the value in the DMAPSA (read in step (s)) and see if it is zero. If it is, go to step (b), otherwise output the following message:

9: RESET ERROR; DATA EXP=XXX DATA REC=XXX

and go to step (b).

SPECIFICATION OF TERMINAL CHARACTERISTICS:

Case 1: Mapped RSX-11M: The RTI is logical unit RTØ:, and communication is accomplished by means of RSX I/O services. Prior to running QSD110, assign the logical unit to a physical terminal number as follows:

>ASN TTn:=RT0:

A PICTURE SYSTEM reset will be issued using H1 to determine the address of the I/O Status Register (IOST). Aside from that, the contents of Host Table locations H1 through H5 are ignored.

Case 2: DL11 Interface, Unmapped Operating System:

H3 = serial interface address
H4 = Ø

Case 3: DZ11 Interface, Unmapped Operating System:

H3 = serial interface address
H4 = 1
H5 = Device Parameters
Bit 12 = Receiver on bit (Must be set)
Bit 11 - Bit 8 = Speed Selection

k. Compare the data read in step (g) with the data sent in step (f). If they are equal, go to step (l), otherwise output the following message:

6: RTI WRITE ERROR; DATA EXP=XXX DATA REC=XXX

l. Write a random number in the DMAPSA using the DIO.
m. Send a DC3 character (octal=23) to initiate an Active Input transfer.
n. Send a count character (octal=377, for a 1 word transfer).
o. Send two characters which represent the PSBUS address of the DMAPSA.
p. Wait to receive two characters from the RTI.
q. Check the two characters received in step (p) against the random number written in step (l) If they are equal, go to step (r). Otherwise, output the following message:

7: RTI READ ERROR; DATA EXP=XXX DATA REC=XXX

r. Write a random number in the DMAPSA with the DIO.
s. Send a DC4 character (octal=24) to cause a PSRESET.
t. Read the DMAPSA via the DIO and see if the data written into it in step (r) is still there. If it has changed, go to step (v), otherwise go to step (u).
u. Increment a timeout counter and check if it has reached a timeout value. If it hasn't, go to step (t), otherwise, output the following message:

8: RTI RESET SEND ERROR; DATA EXP=XXX DATA REC=XXX

and go to step (b).

Error Analysis:

Error #	Prognosis
1:	This error indicates that the data received from the Keyboard/interface did not agree with the data in the table of expected Keyboard codes. If the data coming from the Keyboard is correct, then the problem is in the serial data path.
2:	The RTI could not transfer from its own Active Output Port to its own Passive Input Port. Check the dip switches on the 195177 card and verify that they agree with the port pointed to by the diagnostic. Check the Active Output Port data by examining the 74S374 registers on 195176 locations 60 and 61 to see if the data is getting written and if it is appearing on the PSBUS. If it is, check the 74S374 registers on 195177 locations 60 and 20 to see if the data is being transfered into the RTI Passive Input Port.
3:	The DIO could not transfer to the RTI Pass-Port. This error in conjunction with error #2 should cause the RTI Passive Input Port logic to be examined.
4:	The RTI Active Output Port failed to make the transfer via the DMA. This in conjunction with error #2, should cause the RTI Active Output Port logic to be examined.

Ø = 50 Baud
1 = 75 Baud
2 = 110 Baud
3 = 134.5 Baud
4 = 150 Baud
5 = 300 Baud
6 = 600 Baud
7 = 1200 Baud
10 = 1800 Baud
11 = 2000 Baud
12 = 2400 Baud
13 = 3600 Baud
14 = 4800 Baud
15 = 7200 Baud
16 = 9600 Baud
17 = Reserved

Bit 7 = Odd Parity (Don't care)

Bit 6 = Parity Enable (Must be clear)

Bit 5 = Stop Code (Must be set)

Bit 4 - Bit 3 = Character Length (Must both be set)
Length = 8

Bit 2 - Bit Ø = Line Number for Param Load

Where "n" is the DZ11 Line Number, two common settings for H5 are:

17Ø7n for 9600 Baud

15Ø7n for 2400 Baud

Name: PS2 MULTI-USER REFRESH CONTROLLER REGISTER DIAGNOSTIC

Functional Description:

Successful execution indicates that the Refresh Controller registers can be reset and that they read and write properly. It verifies the following:

a. All of the read write bits in the Refresh Controller registers can be set and cleared.
b. All of the registers in the Refresh Controller reset to the proper state.

This diagnostic consists of two phases.

1. Register Read Write
   Verifies that the Write/Read (W/R) bits of the Refresh Controller registers can be set and cleared.
2. Register Reset
   Verifies that the bits in the Refresh Controller registers go to the proper state in response to PSRESET.

Program Description:

Phase 1: Verifies that the W/R bits of the Refresh Controller can be set and cleared. Program steps are:

a. PSRESET then do the following steps 100 times.
b. Randomly set or clear the Refresh Controller System Interrupt Enable bit and read it back. Check the data that was written with data that came back. If they are the same, continue to step. Otherwise, print the following error message:

```
PHASE 1 DATA ERROR
PSADDRESS=177763 ERROR; DATA EXP=20 DATA REC=0
```

Error #	Prognosis
5:	This error indicates that the RTI tried to write the DMAPSA but the data did not change. This indicates that perhaps the PSA on 195177 locations 33, 43, 53, and 63 is not set properly or that the PSBUS protocol is fouled.
6:	This error indicates that the RTI was able to address the DMAPSA and correctly carry out the PSBUS protocol but that the data was wrong. This should cause a check of the 74S374 on 195176 in locations 60 and 61 and their associated data paths.
7:	This error indicates that the RTI Active Input Port failed to retrieve the data from the DMAPIP properly. This might indicate that the data did not get back to the RTI properly. Check the 74S374 on 195177 locations 60 and 20 and associated data paths. Also check the logic associated with the PSBUS protocol.
8:	This error indicates that the RTI reset command did not influence the DMAPSA. Check the reset logic on 195177 sheet 5, to see if the reset command is being executed.
9:	This command indicates that the DMAPSA did not clear on a reset initiated by the RTI. Check to see that the reset pulse is well formed.

PHASE 2 DATA ERROR
PSADDRESS=177763 ERROR; DATA EXP=0 DATA REC=20

d. Read the RFMNT0 register and if bits 14, 13, 8 and 7 are clear, go to step e. Otherwise, output the following message:

PHASE 2 DATA ERROR
PSADDRESS=177730 ERROR; DATA EXP=0 DATA REC=60600

e. Read the RFDT register and verify that the ECO level of the hardware is in this register. If it is, go to step f. Otherwise, output the following message:

PHASE 2 DATA ERROR
PSADDRESS=177733 ERROR; DATA EXP=0 DATA REC=21

f. Read the RFCLK register and if the clock rate has been set to 16 (60Hz) go to step g. Otherwise, output the following message:

PHASE 2 DATA ERROR
PSADDRESS=177734 ERROR; DATA EXP=16 DATA REC=0

g. Read the RFEXT register and verify that all the bits are clear. If they are, then stop. Otherwise, output the following message:

PHASE 2
PSADDRESS=177735 ERROR; DATA EXP=0 DATA REC=7

NOTE: In the error messages, the numbers shown for PSADDRESS, DATA EXP and REC are examples and may be different when the diagnostic actually reports an error.

c. Write a random number into the RFMNT0 register and read it back. Check bits 14, 13, 8 and 7 in the data that comes back to see if it is the same as the data that was written. If these bits are the same, continue to step d. Otherwise, print the following message:

```
PHASE 1 DATA ERROR
PSADDRESS=177730 ERROR; DATA EXP=60600 DATA REC=0
```

d. Write a random number into the RFCLK register and read it back. Check bits 3-0 in the data that comes back to see if it is the same as the data that was written. If these bits are the same, go to step e. Otherwise, print the following message:

```
PHASE 1 DATA ERROR
PSADDRESS=177734 ERROR; DATA EXP=17 DATA REC=0
```

e. Write a random number into the RFEXT register and read it back. Check bits 2-0 in the data that comes back to see if it is the same as the data that was written. If these bits are the same, stop. Otherwise, print the following message:

```
PHASE 1 DATA ERROR
PSADDRESS=177735 ERROR; DATA EXP=7 DATA REC=0
```

Phase 2: Verifies that the Refresh Controller registers go to the proper state in response to PSRESET. Program steps are:

a. Do the following steps 100 times.
b. Issue a PSRESET.
c. Read the Refresh Controller System Interrupt Enable bit and if it is clear, go to step d. Otherwise, output the following message:

Name: PS2 MULTI-USER REFRESH CONTROLLER MICRO PROGRAM COUNTER DIAGNOSTIC

Functional Description:

Successful execution indicates that the micro program counter operates correctly and that the contents of the micro code proms is proper.

This diagnostic consists of three phases.

1. Single Step
   Verifies that the micro program counter can increment when the MAINT and NOBR bits are set in RFMNT0.
2. ROM Contents
   Verifies that the contents of the ROMS on 195332 and 195333 are correct.
3. PC Exercise
   Verifies that the micro program counter can execute its four basic functions; CALL, RET, JMP and increment.

Program Description:

Phase 1: Verifies that the micro program counter can increment when the MAINT and NOBR bits are set in RFMNT0. Program steps are:

   a. Do a PSRESET.
   b. Write a 60000 into RFMNT0 to enter the nobranch single step mode.
   c. Read the micro program counter then do the following steps 100 times.
   d. Write a 160000 into RFMNT0 to increment the micro program counter and then read it back and verify that it did increment. If it did not, output the following message:

Error Analysis:

Error	Prognosis
Phase 1	This error results when a register does not return the data written into it. Check the addressing logic on the 195331 card if all the bits in a register are bad, and check the individual registers if bits are stuck on or off.
Phase 2	This error results when a register does not contain the value that it should after PSRESET. Check the reset logic associated with the individual registers. The RFDT register will contain a number in it equal to the ECO level of the hardware. If this register cases an error, then the ECO level of the diagnostic does not match the ECO level of the hardware.

g. Verify that the micro program counter went to state 2.

h. Write 140000 into RFMNT0 to step another state.

i. Verify that the micro program counter went to state 1.

j. If states e, g, and i all verified, then go back to state e. Otherwise, output the following message:

```
PHASE 3 DATA ERROR
DATA EXP=XXX DATA REC=XXX
```

If on an error the DATA EXP=XXX field is 3, then the CALL failed. If it is 2 then the RET or increment failed, and if it is 1, then the JMP failed.

PHASE 1 DATA ERROR

DATA EXP=XX DATA REC=XXX

Phase 2: Verifies that the contents of the ROMS on 195332 and 195333 are correct. Program steps are:

a. Do a PSRESET.
b. Write 160000 into RFMNT0 and read back the micro program counter.
c. If the micro program counter is not equal to zero, go to step b.
d. With the micro program counter equal to zero, begin to read the contents of the ROMS. If any location doesn't verify, output the following message:

PHASE 2 DATA ERROR
DATA EXP=XXX,XXX,XXX,XXX DATA REC=XXX,XXX,XXX,XXX

The 195333-950 sheet 3 diagram can be used to determine which PROM or register is failing.

Phase 3: Verifies that the micro program counter can execute its four basic functions; CALL RET,JMP and increment. Program steps are:

a. Do a PSRESET.
b. Write 160000 into RFMNT0 and read back the micro program counter.
c. If the micro program counter is not equal to one , go to step b.
d. With the micro program counter equal to one, write 140000 into RFMNT0 to step one state with branching allowed. Do states e-j 100 times.
e. Verify that the micro program counter went to state 3.
f. Write 140000 into RFMNT0 to step another state.

Identification: QSD113
Date: 27-Jan-78

Name: PS2 MULTI-USER REFRESH CONTROLLER
MICRO CONTROLLER DIAGNOSTIC

Functional Description:

Successful completion of this diagnostic indicates that the Multi-User Refresh Controller arithmetic logic unit is working properly. If verifies the following:

a. The RFDT register can be written and read.
b. The 2901A chips which implement the ALU are functional.
c. The YBUS can be successfully transferred to the DBUS.
d. The flag flip flops are operational.
e. The Refresh Controller can read and write memory.
f. Activity occurs in the Refresh Table.
g. The Refresh Controller can interrupt properly.

This diagnostic consists of 11 phases:

1. PS DATA REGISTER READ WRITE
   Verifies that the RFDT register writes and reads properly.
2. ALU REGISTER READ WRITE
   Verifies that the registers which are internal to the 2901A chips write and read properly.
3. ALU YBUS TO DBUS TRANSFER
   Verifies that data can be transferred from the YBUS to the DBUS on the 195333 card.
4. ALU REGISTER IN ADDRESS
   Verifies the addressing of 16 registers which are internal to the 2901A chips.
5. ALU ARITHMETIC FUNCTIONS
   Verifies that the arithmetic functions of the 2901A chips work properly.
6. ALU LOGIC FUNCTIONS
   Verifies that the logical functions of the 2901A chips work properly.
7. FLAG FLIP FLOPS
   Verifies that the five flag flip flops (FIRST, WB, SKIPSEG, LPSEG and FF1) can be set, cleared and cause the proper branch conditions.

Error Analysis:

Error	Prognosis
Phase 1:	This indicates that the micro program counter could not step with MAINT and NOBR bits set. This indicates that the logic associated with these bits (195332) or the NOBRCH logic on the 195333, may be broken. Also, the maintenance read back path may have problems. When using QSDDT to debug problems associated with the micro program counter, note that the data comes back from the micro program counter in inverted form.
Phase 2:	This indicates that the prom memories that contain the micro program or the pipeline registers on the output of the proms, may be broken. The four words in the error message can be translated into prom locations with the help of the 195333-950 block diagram.
Phase 3:	This error indicates that the micro program counter did not function properly. Check the logic used to generate the TEST signal on the 195332 and the S1 and S0 signals on the 195333.

g. Test the Refresh Controller System Interrupt Request bit and return if it is clear. Otherwise, output the following message:

MAINTENANCE MODE INTERRUPT ERROR

Phase 1: Verifies that the RFDT register writes and reads properly. Program steps are:

a. Call GETM with a parameter of zero to put the Refresh Controller in a mode where the data written into the input side of RFDT is transferred into the output side of RFDT.
b. Do step c 100 times.
c. Write a random number in RFDT and read it back from RFDT. If the number written is different from the one read, output the following message:

PHASE 1 DATA ERROR
DATA EXP=XXX DATA REC=XXX

Phase 2: Verifies that the registers internal to the 2901A chips on the 195333 card, write and read properly. Program steps are:

a. Call GETM with a parameter equal to one to put the Refresh Controller into maintenance mode 0. In this mode, data that is written into RFDT is moved into the 2901A register 0, then from into 1, from 1 into 2, etc., until the data is in register 17. Data is then moved from register 17 into Q and from Q into the output side of RFDT.
b. Do step C 100 times.
c. Write a random number into RFDT and read it back from RFDT. If the number written is different from the one read, output the following message:

10. ALU SHIFT FUNCTIONS
    Verifies that the shift functions of the ALU work properly.
11. MEMORY READ WRITE
    Verifies that the Refresh Controller can write and read Picture System Memory.
12. TABLE ACTIVITY
    Verifies that the Refresh Controller can poll the Task Pointers in the Refresh Table.
13. INTERRUPT AND ERROR BITS
    Verifies that the Refresh Controller can set the Interrupt and Error bits in the Refresh Table.

Program Description:

GETM: This subroutine is called by most of the phases in this diagnostic to get into maintenance mode. Program steps are:

a. Do a PSRESET.
b. Write -1 into RFDT and acknowledge the Refresh Controller System Interrupt Request bit to get the Refresh Controller into maintenance mode.
c. Read the Refresh Controller System Interrupt Request bit and if it is set, go to step d. Otherwise output the following message:

DIDN'T GET INTO MAINTENANCE MODE

d. Test the input parameter and if it is zero, then return. Otherwise, continue to step e.
e. Subtract one from the input parameter and write the result in RFDT. This is used as the maintenance function code and causes the activity described in Phases 2-10.
f. Acknowledge the Refresh Controller System Interrupt Request bit to cause the Refresh Controller to execute the code in RFDT.

c. Write a random number in RFDT and read it back. If the low order four bits of the number written differ from the number read, output the following message.

```
PHASE 4 DATA ERROR
DATA EXP=XXX DATA REC=XXX
```

Phase 5: Verifies the arithmetic functions of the 2901A chips. Program steps are:

a. Call GETM with a parameter equal to four to put the Refresh Controller into maintenance mode 3. In this mode the data in RFDT is manipulated by the arithmetic functions and if everything is in order, it is written back into RFDT.
b. Do step c 100 times.
c. Write a random number into RFDT and read it back. If the number written is different from the number read, output the following message:

```
PHASE 5 DATA ERROR
DATA EXP=XXX DATA REC=XXX
```

Phase 6: Verifies the logical functions of the 2901A chips. Program steps are:

a. Call GETM with a parameter equal to five to put the Refresh Controller into maintenance mode 4. In this mode, the data in RFDT is manipulated by the logical functions, and if everything is in order, is written back into RFDT.
b. Do step c 100 times.
c. Write a random number into RFDT and read it back. If the number written is different from the number read, output the following message:

PHASE 2 DATA ERROR
DATA EXP=XXX DATA REC=XXX

Phase 3: Verifies that the data can be transferred from the YBUS to the DBUS on the 195333 card. Program steps are:

a. Call GETM with a parameter equal to two to put the Refresh Controller into maintenance Mode 1. In this mode data that is written into the input portion of RFDT is moved into a 2901A register. This register is enabled onto the YBUS and the YBUS is driven onto the DBUS. The data from the DBUS is stored in another register and moved from there onto the output portion of RFDT.

b. Do step c 100 times.

c. Write a random number in RFDT and read it back. If the number written is different from the one read, output the following message:

PHASE 3 DATA ERROR
DATA EXP=XXX DATA REC=XXX

Phase 4: Verifies the addressing of the internal registers of the 2901A chips. Program steps are:

a. Call GETM with a parameter equal to three to put the Refresh Controller into maintenance mode 2. In this mode the micro code writes the numbers 0-17 into the 2901A registers 0-17 (data in address). Data is then taken from RFDT and placed in the Q register. The low order four bits select one of the 16 registers and the contents of that register is written into RFDT.

b. Do step c 100 times.

PHASE 6 DATA ERROR
DATA EXP=XXX DATA REC=XXX

Phase 7: Verifes that the five flag flip flops (FIRST, WB, SKIPSEG, LPSEG and FF1) can be set, cleared and cause the proper branch conditions. Program steps are:

a. Call GETM with a parameter equal to six to put the Refresh Controller into maintenance mode 5. In this mode, the micro program tests all five flip flops, and if they are operational the data in the input side of RFDT is moved to the output side of RFDT.
b. Do step c 100 times.
c. Write a random number into RFDT and read it back. If the number written is different from the number read, output the following message:

PHASE 7 DATA ERROR
DATA EXP=XXX DATA REC=XXX

Phase 10: Verifies that the ALU shift functions work properly. Program steps are:

a. Call GETM with a parameter equal to seven to put the Refresh Controller in maintenance mode 6. In this mode, data is read from RFDT and is shifted and rotated inside the 2901A chips. If there are no failures in this logic, the data will be unmodified when it is written back into RFDT.
b. Do step c 100 times.
c. Write a random number into RFDT and read it back. If the number written differs from the number read, output the following message:

PHASE 10 DATA ERROR
DATA EXP=XXX DATA REC=XXX

Phase 11: Verifies that the Refresh Controller can read and write Picture System memory. Program steps are:

a. Call GETM with a parameter equal to eight to put the Refresh Controller in maintenance mode 7. In this mode, the data in Picture System memory location 0 is read, incremented and written back into location 1. Location 1 is then read, incremented and written back into location 2. This continues until the highest address in memory (177377). When this address is written, the process starts over.

b. Do steps c 100 times.

c. Write a random number in location zero and poll a random location until it changes. If it does not change after a long time, output the following message:

```
PHASE 11 TIMEOUT ERROR
PSADDRESS=XXX
```

If the data does change, verify that the contents of the address is equal to the value of the address plus the random number written into location zero. If this is not true, output the following message:

```
PHASE 11 DATA ERROR
PSADDRESS=XXX DATA EXP=XXX DATA REC=XXX
```

Phase 12: Verifies that the Refresh Controller can poll the Task Pointers in the Refresh Table. Program steps are:

a. Do steps b-f 100 times.

b. Do a PSRESET and write a Refresh Table into Picture System memory.

c. Verify that the PSRESET caused the Refresh Controller System Interrupt Request bit to set. If it did not, output the following message:

PHASE 12 INTERRUPT ERROR
DATA EXP=20 DATA REC=0

d. Write the address of the first location of the Refresh Table into RFDT and acknowledge the Refresh Contorller System Interrup Request.
e. Verify that the Refresh Controller System Interrupt Request bit remained clear after it was acknowledged. If it did not, output the following message:

PHASE 12 INTERRUPT ERROR
DATA EXP=0 DATA REC=20

f. Read the address of the current task Pointer from the Refresh Table. If it does not change after a time out, output the following message:

PHASE 12 TIMEOUT ERROR

Phase 13: Verifies that the Refresh Controller can set the Interrupt and Error bits in the Refresh Table. Program steps are:

a. Do a PSRESET and write a Refresh Table with 32. 177775 Task Pointers.
b. Do steps c for all 32 task pointers.
c. Acknowledge the Refresh Controller System Interrupt Request bit and wait for it to set again. If after a time out it doesn't set, output the following message:

PHASE 13 INTERRUPT ERROR

d. Read the first four locations in the Refresh Table and verify the appropriate Interrupt and Error bits are set. If they are not, output the following message:

```
PHASE 13 DATA ERROR
XXX/  XXX      XXX
XXX/  XXX      XXX  *
XXX/  XXX      XXX
XXX/  XXX      XXX
```

This error message shows the Picture System address, contents of that address, expected contents of that address and an error mark according to the following format:

ADDRESS/CONTENTS	EXPECTED	ERRORMARK
XXX/XXX	XXX	*

Error Analysis:

DIDN'T GET INTO MAINTENANCE MODE - This error indicates that the Refresh Controller System Interrupt Request bit did not set upon entrance into maintenance mode. Check the 195331 logic associated with the interrupt bit and the RFDT register on the 195333 card.

MAINTENANCE MODE INTERRUPT ERROR - This error condition indicates that after the Refresh Controller has been dispatched to one of the eight maintenance modes, it set the Refresh Controller System Interrupt Request bit. This occurs when the interrupt bit on the 195331 card has failed.

PHASE 1 DATA ERROR - This error indicates that the data could not be transferred from the input side of RFDT through the 2901A chips to the output side of the RFDT. This might be caused by bad 2917 or 2901A chips on the 195333 card.

PHASE 2 DATA ERROR - This error condition occurs if the internal registers of the 2901A chips are bad. Check the A3-A0 and B3-B0 inputs the the chips. If they are active consider replacing the offending 2901A chip.

PHASE 3 DATA ERROR - This message is output when the data path from the YBUS to the DBUS (see 195333-950) does not work properly. Check the 74S241 chips on the 195333 card.

PHASE 4 DATA ERROR - This message indicates that the addressing of the 16 registers internal to the 2901A is not proper. Check the A3-A0 and B3-B0 inputs to the chips. If they are active, consider replacing the offending 2901A chip.

PHASE 5 DATA ERROR & PHASE 6 DATA ERROR - These messages indicate that the arithmetic and logical functions in the 2901A chips did not operate correctly. If the I5-I3 signals are active, then the 2901A chips on the 195333 card should be suspect.

PHASE 7 DATA ERROR - This message appears if the flag flip flops are not operating correctly. The micro program returns an error code to indicate which flip flop caused the trouble. This error code appears in the DATA REC=XXX field and the following table explains each code.

ERROR CODE	ERROR CONDITION
1	FIRST flip flop didn't set
2	FIRST flip flop didn't clear
3	WB flip flop didn't set
4	WB flip flop didn't clear
5	SKIPSEG flip flop didn't set
6	SKIPSEG flip flop didn't clear
7	LPSEG flip flop didn't set
10	LPSEG flip flop didn't clear
11	FF1 flip flop didn't set
12	FF1 flip flop didn't clear

If one of these errors occur check to see if the state of the flip flop does go up and down. If it does, check the TS3-TS0 micro bits to insure that they are active. Also check the TEST and INV signals. (All these signals are on the 195332 card).

PHASE 10 DATA ERROR - This error occurs when the ALU shift functions do not work properly. Check the RAM3 and Q3 signals on the 2901A chips, the I8-I0 inputs for activity.

PHASE 11 TIMEOUT ERROR - This error indicates that a location in Picture System memory could not be written by the Refresh Controller. Inspect the 75S374 chips on the 195333 card which implement the PSA. Also look at the write control logic on the 195332 card.

PHASE 11 DATA ERROR - This error indicates that the location got written with the wrong data. This could be caused by a bad PSA (195333) or bad Picture System Memory.

PHASE 12 INTERRUPT ERROR - This message is output to indicate that Refresh Controller System Interrupt Request bit did not set in response to PSRESET. Inspect the 195331 interrupt logic.

PHASE 12 TIMEOUT ERROR - This message indicates that the polling of the Refresh Table did not occur normally. Check the clock logic on the 195331 card.

PHASE 13 DATA ERROR - This error condition indicates that Refresh Controller could not read the Refresh Controller System Interrupt Request properly. Check the INTREQ signal on the 195332 card for activity.

Name: PS2 MULTI-USER REFRESH CONTROLLER
REFRESH FUNCTION DIAGNOSTIC

Functional Description:

Successful execution of this diagnostic indicates that the Refresh Functions of the Multi-User Refresh Controller are operational. This diagnostic consists of seven phases:

1. INITIALIZE REFRESH
   Verifies that the Picture System Memory can be initialized for use as a display list.
2. ADD SEGMENT 1
   Verifies that a segment can be added to a display list.
3. COMPACT SEGMENT
   Verifies that the available memory in the display list be collected and placed at the end of a segment.
4. REPLACE SEGMENT AND COMPACT
   Verifies that one segment can be replaced by another segment and that after this replacement, all available memory in the display list can be collected and placed after the end of the new segment.
5. REPLACE SEGMENT
   Verifies that one segment can be replaced by another.
6. CHANGE STATUS OF SEGMENT
   Verifies that the Refresh Controller can search for a segment name and change its status according to an input parameter.
7. DELETE SEGMENT AND COMPACT
   Verifies that the Refresh Controller can delete a segment and collect all the available memory and place it where the old segment used to start.

Program Description:

Phase 1: Verifies that the Picture System Memory can be initialized for use as a display list. Program steps are:

a. Do steps b-e 100 times.
b. Do a PSRESET and build a Refresh Table in Picture System memory.
c. Use a Task Command & Parameter Table to invoke the INITIALIZATION command.
d. If after a time out period, the Refresh Controller System Interrupt Request bit does not set, output the following message:

PHASE 1 INTERRUPT ERROR

e. After the Refresh Contoller System Interrupt Request bit sets, verify that the display list is properly initialized. If it is not, output the following message:

PHASE 1 DATA ERROR
XXX/XXX XXX *
.
.
.

Phase 2: Verifies that the ADD command can be used to add a segment to the display list. Program steps are:

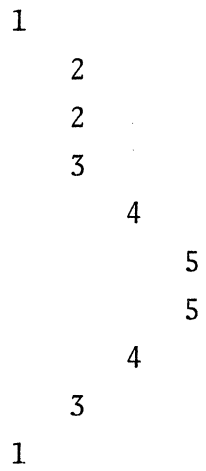
a. Do steps b-e 100 times.
b. Do a PSRESET and build a Refresh Table in Picture System memory.
c. Use a Task Command & Paramter Table to invoke the ADD command.
d. Wait for the Refresh Controller System Interrupt Request bit to set and output the following message if it does not.

PHASE 2 INTERRUPT ERROR

e. Verify that the segment has been added to the display list. If it has not, output the following message:

```
PHASE 2 DATA ERROR
XXX/XXX    XXX    *
.
.
.
```

f. Wait several seconds before continuing to allow the user to verify that Line Generator is displaying the following image.



Phase 3: Verifies that the COMPACT command collected all the available memory and placed it at the end of a segment. Program steps are:

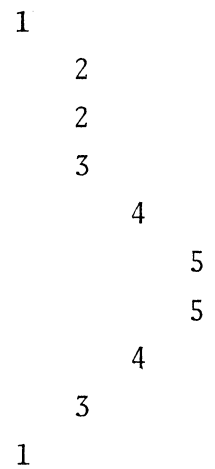
a. Do steps b-e 100 times.
b. Do a PSRESET and build a Refresh Table in Picture System Memory.
c. Use a Task Command & Parameter Table to invoke the COMPACT command.
d. Wait for the Refresh Controller System Interrupt Request bit to set and output the following message if it does not:

```
PHASE 3 INTERRUPT ERROR
```

e. Verify that the compaction has occured and it has not output the following message:

```
PHASE 3 DATA ERROR
XXX/XXX    XXX    *
.
.
.
```

f. Wait several seconds before continuing to allow the user to verify that the Line Generator is displaying the following image.



Phase 4: Verifies that the REPLACE/COMPACT command replaces one segment with another and collects all the available memory in the display list and places it after the new segment. Program steps are:

a. Do steps b-e 100 times.
b. Do a PSRESET and build a Refresh Table in Picture System Memory.
c. Use a Task Command & Parameter Table to invoke the REPLACE/COMPACT command.
d. Wait for the Refresh Controller System Interrupt Request bit to set and output the following message:

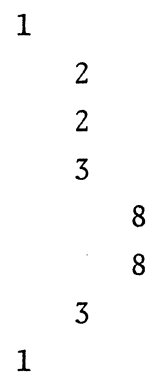
```
PHASE 4 INTERRUPT ERROR
```

e. Verify that the old segment was replaced and that the available memory was collected and

placed after the end of the new segment. If it was not, output the following message:

```
PHASE 4 DATA  ERROR
XXX/XXX    XXX    *
```

f. Wait several seconds before continuing to allow the user to verify that the Line Generator is displaying the following image.



Phase 5: Verifies that the REPLACE command replaces one segment with another. Program steps are:

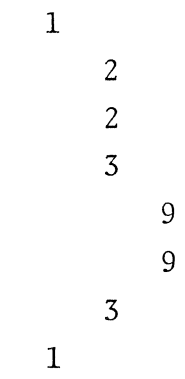
a. Do steps b-e 100 times.
b. Do a PSRESET and build a Refresh Table in Picture System Memory.
c. Use a Task Command & Parameter Table to invoke the REPLACE command.
d. Wait for the Refresh Controller System Interrupt Request bit to set and output the following message if it does not.

```
PHASE 5 INTERRUPT ERROR
```

e. Verify that the old segment was replaced with the new segment and if it wasn't, output the following message:

```
PHASE 5 DATA ERROR
XXX/XXX    XXXX    *
.
.
.
```

f. Wait several seconds before continuing to allow the user to verify that the Line Generator is displaying the following image.



Phase 6: Verifies that the CHANGE STATUS command can change the status of a segment name. Program steps are:

a. Do steps b-e 100 times.

b. Do PSRESET and build a Refresh Table in Picture System Memory.

c. Use a Task Command & Parameter Table to invoke the CHANGE STATUS command. Use the four low order bits in the counter from step a as the parameter word.

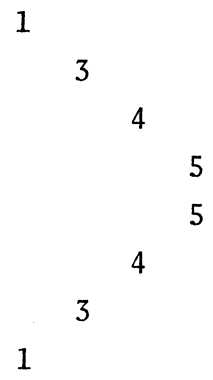
d. Wait for the Refresh Controller System Interrupt Request bit to set and output the following message if it does not:

```
PHASE 6 INTERRUPT ERROR
```

e. Verify that the status of the segment changed according to the Parameter word. If it did not, output the following message:

```
PHASE 6 DATA ERROR
XXX/XXX    XXX    *
.
.
.
```

f. Wait several seconds before continuing to allow the user to verify that the Line Generator is displaying the following image:



Phase 7: Verifies that the DELETE/COMPACT command can delete a segment and collect all the free memory in the display list and leave it in the place of that segment. Program steps are:

a. Do steps b-e 100 times.

b. Do a PSRESET and build a Refresh Table in Picture System Memory.

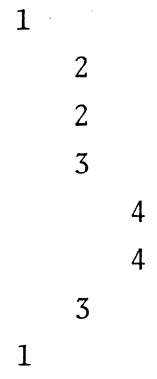
c. Use a Task Command & Parameter Table to invoke the DELETE/COMPACT command.

d. Wait for the Refresh Controller System Interrupt Request bit to set and output the following message if it does not.

```
PHASE 7 INTERRUPT ERROR
```

e. Verify that the segment was deleted and that the available memory in the display list was collected and left in its place. If this did not occur, output the following message:

f. Wait several seconds before continuing to allow the suer to verify that the Line Generator is displaying the following image:



Error Analysis:

The errors that this diagnostic is likely to uncover are more easily diagnosed with the QSD111, QSD112 and QSD113 diagnostics. The phase 2-7 generate pictures and the user should verify these pictures. If no pictures appear, check the SKIPSEG signals on the 195331 and 195332 cards.

Date: 20-Jan-78

Name: PS2 MULTI-USER REFRESH CONTROLLER
REFRESH ERROR DIAGNOSTIC

Functional Descriptions:

This diagnostic invokes the various Refresh Controller commands with bad parameters. It verifies that the Refresh Controller can detect these bad parameters and report the errors.

This diagnostic consists of five phases:

1. INITIALIZE REFRESH
   Initializes the display list with too little memory and verifies that this causes an error.
2. COMPACT SEGMENT
   Verifies that when the COMPACT command fails to find the proper segment an error is generated.
3. REPLACE SEGMENT AND COMPACT
   Verifies that when the REPLACE/COMPACT command fails to find the proper segment, an error is generated.
4. CHANGE STATUS OF SEGMENT
   Verifies that when the CHANGE STATUS command fails to find the proper segment, an error is generated.
5. DELETE SEGMENT AND COMPACT
   Verifies that when the DELETE/COMPACT command fails to find the proper segment, an error is generated.

Program Description:

Phase 1: Verifies that the INITIALIZE command will cause an error when too little memory is allocated. Program steps are:

a. Do steps b-e 100 times.
b. Do a PSRESET and build a Refresh Table in Picture System Memory.
c. Use the Task Command & Parameter Table to invoke the INITIALIZATION command with too little memory.

d. Wait for the Refresh Controller System Interrupt Request bit to set and output the following message if it does not.

PHASE 1 INTERRUPT ERROR

e. Verify that the command word in the Task Command & Parameter Table has been changed from 1 to 100100. If it has not, output the following message:

PHASE 1 DATA ERROR
1000/XXX 100100 *

Phase 2: Verifies that when the COMPACT command fails to find the proper segment name, an error is generated. Program steps are:

a. Do steps b-e 100 times.
b. Do a PSRESET and build a Refresh Table in Picture System Memory.
c. Use the Task Command & Parameter Table to invoke the COMPACT command with a non-existant segment name.
d. Wait for the Refresh Controller System Interrupt Request bit to set and output the following message if it does not.

PHASE 2 INTERRUP ERROR

e. Verify that the command word in the Task Command & Parameter Table has been changed from 5 to 100241. If it has not, output the following message:

PHASE 2 DATA ERROR
1000/XXX 100241 *

Phase 3: Verifies that when the REPLACE/COMPACT command fails to find the proper segment name, an error is generated. Program steps are:

a. Do steps b-e 100 times.
b. Do a PSRESET and build a Refresh Table in Picture System Memory.
c. Use the Task Command & Parameter Table to invoke the REPLACE/COMPACT with a non-existant segment name.
d. Wait for the Refresh Controller System Intertupt Request bit to set, and if it does not, output the following message:

```
PHASE 3 INTERRUPT ERROR
```

e. Verify that the command word in the Task Command & Parameter Table has been changed from 13 to 100540. If it has not, output the following message:

```
PHASE 3 DATA ERROR
1000/XXX    100441    *
```

Phase 4: Verifies that when the CHANGE STATUS command fails to find the proper segment name, an error is generated. Program steps are:

a. Do steps b-e 100 times.
b. Do a PSRESET and build a Refresh Table in Picture System Memory.
c. Use the Task Command & Parameter Table to invoke the CHANGE STATUS command with a non-existant segment name.
d. Wait for the Refresh Controller System Interrupt Request bit to set and if it does not, output the following message:

```
PHASE 4 INTERRUPT ERROR
```

e. Verify that the command word in the Task Command & Parameter Table has been changed from 14 to 100601. If it has not, output the following message:

```
PHASE 4 DATA ERROR
1000/XXXX    100541    *
```

Phase 5: Verifies that when the DELETE/COMPACT command fails to find the proper segment name an error is generated. Program steps are:

a. Do steps b-e 100 times.
b. Do a PSRESET and build a Refresh Table in Picture System Memory.
c. Use the Task Command & Parameter Table to invoke the DELETE/COMPACT command with a non-existant segment name.
d. Wait for the Refresh Controller System Interrupt Request bit to set and if it does not, output the following message:

```
PHASE 5 INTERRUPT ERROR
```

e. Verify that the command word in the Task Command & Parameter Table has been changed from 10 to 100401. If it has not, output the following message:

```
PHASE 5 DATA ERROR
1000/XXX     100341    *
```

Error Analysis:

The errors that this diagnostic is likely to uncover are more easily diagnosed with the QSD111, QSD112 and QSD113 diagnostics.

Name: PS2 MULTI-USER REFRESH CONTROLLER
DEVICE CONTROL DIAGNOSTIC

Functional Description:

Successful execution of this diagnostic indicates that the Refresh Controller device control functions operate correctly. It verifies the following:

a. The device polling commands work properly.
b. The Cursor and Light Pen commands work properly.
c. The various Picture System devices work together without conflict.

This diagnostic consists of two phases.

1. DEVICE POLLING
   Verifies that all the DEVICE POLLING commands work properly.
2. ALL DEVICES
   Give the user the capability to verify that the Light Pen, Tablet and other Picture System devices work together properly.

Program Description:

Phase 1: Verifies that all DEVICE POLLING commands work properly.
Program steps are:

a. Do PSRESET.
b. Set up a device polling command table which uses the DMAPSA.
c. Write the following sequence of numbers into the DMAPSA: 0, 1, 0, 2, 40000, 4, 10, 40, 20 and 4. After each number, verify that the Refresh Controller made one of the following replacements: 0, 1, 400, 1000, 20000, 2000, 10, 4000, 10020 and 10020.
d. Do step c 100 times. If the Refresh Controller responds with the wrong data, output the follow

ing message:

```
PHASE 1 DATA ERROR
DATA EXP=XXX DATA REC=XXX
```

Phase 2: Gives the user the capability to verify that the Light Pen, Tablet and other Picture System devices work together properly.

a. Do a PSRESET and set up a Refresh Table.
b. Poll the user waiting for input. At this point the program expects a character from the terminal to indicate which device is to be tested.
c. After the user enters a device name the program sets up device polling commands and display lists to allow the state of the various devices to appear on the screen.
d. The LIGHT PEN and TABLET put up cursor symbols and the other devices put the contents of their data registers in binary on the screen.
e. The program exits when the user types X.

Error Analysis:

Error	Prognosis
Phase 1	This error results when a device polling command fails to operate correctly. This may indicate that the address and data paths on the Refresh Controller are failing.

Identification: QSD120
Date: 3-Jan-79

Name: PS2 DYNAMIC MEMORY TEST
USING MAP ACTIVE I/O

Functional Description:

This Picture System Memory Test is designed to detect dynamic failures which may not be detected by QSD002. It runs considerably faster than QSD002, and offers various user options, including (a) user specification of memory test address range, (b) automatic determination of available memory, (c) user selection of DIO or DMA data transfer, (d) user specification of data pattern, and other options.

The test consists of three phases:

1. Address Data
2. Zero/Ones Test (5 passes)
3. Random Data (5 passes)

Program Description:

The following description pertains to all three phases, with the exception of information which is specific to each phase, as described in steps p through r:

a. First time only: if P101 is non-zero, announce the user specified memory range. Otherwise, determine the available memory, and announce the detected range. The available memory is determined as follows: write 0 and 52525 into locations 0 and 1 respectively via DIO, then read and verify location 0. Write 40000 and 52525 into locations 40000 and 40001, and verify location 40000. Proceed in this fashion with locations 100000 and 140000. The first error determines the amount of available memory, always starting at 0 and extending to a maximum of 177377.

b. First time only: if P77 equals zero, use DIO for block data transfers; otherwise, use DMA. Set a flag and announce whether DIO or DMA is being used.
c. First time only: if P76 is non-zero, interrogate the user for options to specify random key, block increment, or user data sequence.
d. First time each phase: Reset the Picture System and announce the phase.
e. Load the source data buffer with data for this phase and this pass, as described in steps p through r.
f. Output the source data buffer to the first memory block (normally 0 through 776) via DIO or DMA.
g. If DMA is disabled, go to step j.
h. Start the data from the first block into a destination buffer via DMA, no wait. There are two destination buffers, and input data is double buffered to allow overlapping of input data transfer and verification, provided DMA is enabled.
i. Start the MAP transferring data from the first memory block to the second block.
j. Wait for MAP done, reset the MAP, and initialize the MAP to read data from block n via MAP active input, and write the identical data in block n + 1 via MAP active output. Start the MAP.
k. Start the data from block n into a destination buffer via DIO or DMA. If DMA is enabled, do not wait for completion. If DMA is disabled, of course, the program is occupied controlling the DIO data transfer.
l. Compare the last block read in with the source data block. The verification subroutine is described in steps s through y.
m. If an error has occurred, go to step f, interpreting "first memory block" as the block following the one in which an error was discovered.

n. If the last block has not been verified, increment n and go to step j.

o. If there are more passes in this phase, go to step e; otherwise exit the phase.

p. Phase one consists of one pass using address data or a user selected data pattern. Address data is not actually the address of each location, but data intended to confirm that all address lines are functioning. The first location in the block contains 0, and subsequent locations are incremented by octal 401. The block size is octal 776, and the last location of the block contains 177375. The second block normally begins with 776 through 1000 loaded with 0, 401, 1002. The distance between blocks is controlled by the Block Increment parameter which must be greater than or equal to the block size, and is normally 776. The data is organized in this fashion so that, for example, locations 0 and 1000, which differ in address by one bit, will not contain the same data in either 8-bit slice. Alternatively, if the user specifies, for example, a data sequence of length 3 consisting of 52525, 125252, 177777, the sequence is replicated throughout the length of the source data block.

q. Phase two consists of zero/one data in 5 passes. The data sequences are of length 2 consisting of:

    pass 1: 0, 0
    pass 2: 125252, 52525
    pass 3: 177777, 177777
    pass 4: 52525, 125252
    pass 5: 0, 0

r. Phase three consists of 5 passes using new random data for each pass. Normally the random key is initialized to zero, but the user may specify some other initial value.

s. Steps s through y describe the verification subroutine. As input data is double-buffered, so is verification. Select the current input buffer for verification, and toggle the selector.

t. Step through the source buffer and destination buffer, comparing expected and received values. If an error occurs, go to step u. Exit the verification subroutine when done.

u. Set the error flag, and compute the Picture System address corresponding to the erroneous data in the destination buffer. Read the Picture System Memory location again via DIO, and compare with the expected data in the source buffer. If the data matches, set error index x = 1 and go to step x.

v. Read the source location for the MAP active data transfer (e.g. if an error occurs in location 776, the data was transferred by the MAP from location 0 to location 776, so check location 0). If this data agrees with the expected data, set index x = 2 and go to step x (this is the normal index for memory failures).

w. Set index x = 3.

x. Print the following error message:
MEMORY ERROR; PSADR = nnnnnn EXPT = nnnn RECD = nnnn
BLK INCREMENT = nnn INDEX = x

y. Go to step t to continue checking the block.

User Options:

a. If P77 is non-zero, DMA is enabled. Default P77 = 0.

b. If P101 = 0, the program determines the amount of available memory. Note that a hard memory failure in location 0, 40000, 100000, or 140000 will cause incorrect determination of memory size. The user should verify the amount of available memory. If P101 is non-zero, P100 is the inclusive lower bound and P101 is the exclusive upper bound of the area of memory tested. Default P101 = 0.

c. If P76 is non-zero, the program will interrogate the user after the "X" command to allow him to specify the random key for phase 3 ("R"), the block increment parameter for all phases ("I"), or the data sequence for phase 1 ("D"). Default P76 = 0. For more information, see program steps p and r.

Error Prognosis:

There is one error message with three possible error indices:

INDEX = 1 The data stored in memory was found to be correct using a single-word read via DIO. However, a previous block read via DIO or DMA incurred an error. This may indicate a transient memory output or Picture System interface problem.

INDEX = 2 This index indicates the most probable type of error, a normal memory failure. To relate address and bit to an IC on a 195143 card, see the prognosis for error no. 3 in QSD002.

INDEX = 3 Data was stored in a memory location and verified once, and subsequently went bad. Again, see the prognosis for error no. 3 in QSD002.

5.1.6 DESCRIPTIONS OF ACCEPTANCE TEST PROGRAMS

This section contains detailed descriptions of programs used for acceptance tests and readiness testing.

Name: MAP Speed Tests

Functional Description: This program is used in conjunction with an oscilloscope to verify the processing time required for the following Matrix Arithmetic Processor commands:

PHASE 1/11	Matrix concatenation
PHASE 2/12	Characters, and Line Generator Status
PHASE 3/13	Dot clipped, and Dot not clipped
PHASE 4/14	Line not clipped
PHASE 5/15	Line clipped in 1 plane
PHASE 6/16	Line clipped in 2 planes
PHASE 7/17	Line clipped, non-visible
PHASE 10/20	MAP No-op

Phases 1 through 10 load data into PS Memory and utilize MAP Active Input. Phases 11 through 20 transmit data to the MAP Passive Input Port via DMA.

Program Description:

a. If not first pass for this phase, go to step g.

b. Output a message announcing the phase, and reset the Picture System.

c. Clear MAP halt, and disable MAP output.

d. Load the data buffer with the hit window data, and output the data via Direct I/O.

e. Load the data buffer with the data for this phase.

f. If phase 1-10, output the data to Picture System Memory beginning at location Ø via DMA, and start MAP Active Input. For these phases, there is a "jump to PSMEM Ø" command at the end of the buffer.

g. Initialize the phase counter.

h. Wait for real-time clock.

i. If phase 11-20, output the data buffer to the MAP via DMA.

j. Decrement and test the phase counter. If non-zero, go to step h, otherwise exit.

Identification: RSD001
Date: 10/31/78

Name: MAP Function Tests (2D,3D,4D)

Functional Description: This program visually demonstrates MAP processing of two-dimensional, three-dimensional, and four-dimensional data. The x and y data for all three phases is identical, and the z data for phases 2 and 3 is identical. The three three phases are:

PHASE 1	2-D Data	(two squares with connected vertices.)
PHASE 2	3-D Data	(frustrum)
PHASE 3	4-D Data	(via manipulation of the homogeneous coordinate w, the frustrum of phase 2 is converted to a cube.)

Program Description:

a. If not first pass for this phase, go to step d.

b. Announce the phase, and reset the Picture System.

c. Initialize the MAP: clear MAP halt, enable MAP Active Output, and direct MAP output to the Line Generator.

d. Load the data buffer and output data to the MAP via DMA to load a perspective window matrix.

e. Load the data buffer with the data for the present phase, and initialize the phase counter.

f. Wait for real-time clock.

g. Output data for this phase to the MAP via DMA.

h. Output via DMA to the MAP a matrix concatenation command and rotation matrix to rotate the object 1/120th revolution about the x-axis.

i. Decrement and test the phase counter. If non-zero, go to step f, otherwise exit.

j. The octal data for the three phases is summarized below:

RSR (2D/3D/4D, Absolute, MDD,12)

X (ALL PHASES)	Y (ALL PHASES)	Z (PH 2,3)	W (PH 3)
0	0	0	20000
10000	0	0	20000
10000	10000	0	20000
0	10000	0	20000
0	0	0	20000
0	0	20000	40000
20000	0	20000	40000
20000	20000	20000	40000
0	20000	20000	40000
0	0	20000	40000

RSR (2D/3D/4D, Absolute, MDMD,6)

10000	0	0	20000
20000	0	20000	40000
10000	10000	0	20000
20000	20000	20000	40000
0	10000	0	20000
0	20000	20000	40000

Identification: RSD002
Date: 10/31/78

Name: DMA Timing and Simultaneous Direct I/O Test

Functional Description: Phases 1 through 3 are used in conjunction with an oscilloscope to verify the DMA data transfer rate. Phases 4 through 6 are used to demonstrate, in conjunction with an oscilloscope, simultaneous DMA and Direct I/O data transfers.

Program Description:

a. If not the first pass for this phase, go to step c.

b. Announce the phase.

c. Announce the phase, and load the data buffer with MAP no-ops for phase 1, 2, 4, or 5, or Line Generator status commands for phase 3 or 6.

d. Direct the DMA to Picture System Memory for phase 1 or 4, the MAP passive input port for phase 2 or 5, or the Line Generator passive input port for phase 3 or 6. Initialize the phase counter.

e. Wait for real-time clock. This causes a series of DIO read operations, as the real-time clock interrupt request is repeatedly sampled until it goes high.

f. Initiate a DMA data transfer to the selected destination. If phase 1-3, wait for DMA completion.

g. Decrement and test the phase counter. It non-zero, go to step e, otherwise exit.

Identification: RSD003
Date: 10/31/78

Name: Refresh Modes Test (Single-User Refresh Controller)

Functional Description: This program causes the display of refresh data in three modes:

PHASE 1 Single-Buffered Autorefresh
PHASE 2 Double-Buffered Software Initiated Refresh
PHASE 3 Segmented Autorefresh

Program Description:

Phase 1: Single-Buffered Autorefresh

a. If not first pass for this phase, go to step d.

b. Announce the phase, reset the Picture System, initialize the refresh clock to run at 30Hz, and load Picture System Memory with a pattern of zig-zag lines.

c. Initialize Refresh Start Address and Refresh Limit, and start Autorefresh.

d. Initialize the phase counter.

e. Wait for real-time clock.

f. Decrement and test the phase counter. If non-zero, go to step e, otherwise exit.

Phase 2: Double-Buffered Software Initiated Refresh

a. If not first pass for this phase, go to step d.

b. Announce the phase, reset the Picture System, and set the real-time clock rate to 120Hz.

c. Load two Picture System Memory buffers for refresh patterns which constitute interlaced strings of short zig-zag vectors. Displaying the two buffers in double-buffered mode will create the visual effect of a grid.

d. Initialize the phase counter.

e. Wait for real-time clock. If phase counter is zero, exit; otherwise decrement the counter.

f. Wait for RFSTOPPED.

g. Switch refresh buffers by loading alternate values into Refresh Start Address and Refresh Limit.

h. Start autorefresh by setting RFSTART.

i. Go to step e.

Phase 3: Segmented Autorefresh

The display buffer for this phase consists of five segments, a fixed segment A to initialize the Line Generator, and four segments B through E, each of

which draws a small box with a distinctive pattern inside. The positions of the four segments are rotated on each frame, creating the visual effect of four boxes each containing the sum of the patterns in B through E. If the initial sequence of segments is ABCDE in the display buffer, subsequent sequences are ACDEB, ADEBC, etc.

a. If not first pass for this phase, go to step d.

b. Reset the Picture System. Set the real-time clock rate at 60Hz. Announce the phase.

c. Write the fixed segment A into Picture System memory.

d. Initialize the phase counter.

e. Update the index which sequentially selects segment B,C,D, or E.

f. Load data for the selected segment into the data buffer. Wait for real-time clock.

g. Update the refresh buffer by performing steps h through q. Decrement and test the phase counter. If non-zero, go to step e, otherwise exit.

h. Steps i through q describe the procedure for deleting a segment if it is already in the refresh buffer, then writing it as the last segment in the refresh buffer (subroutine WTSG).

i. HALT Autorefresh. Write the name of the new segment and data for the segment at the current end of the refresh buffer (Refresh Input Limit), via DMA.

j. Wait for RFSTOPPED. Set up to search for the name of the new segment in "exact match" mode, to determine whether the segment name is already in use within the refresh buffer. Set RFSTART to start the Refresh Controller.

k. Wait for RFSTOPPED or RFHOLD. Bump the refresh input limit and refresh writeback limit to the end of the new segment.

l. If RFSTOPPED, there is no old segment to delete, so go to step q.

m. Set the writeback address equal to the address of the old segment, causing it to be deleted.

n. Set Refresh Status to: search, mode = Ø (match any segment name), skipseg, match hold, match DEC. Acknowledge the match request. This skips to the end of the old segment.)

o. Wait for RFHOLD. Set Refresh Status to: Skipseg, Writeback. Acknowledge the match request. (This reads the remainder of the refresh buffer, including the new segment, in writeback mode, compacting the refresh buffer and deleting the old segment.)

p. Wait for RFSTOPPED. Decrease the Refresh Limit to compensate for compaction.

q. Start autorefresh. Exit.

Identification: RSD004
Date: 11/3/78

Name: Character Generator / Line Generator Speed Tests

Functional Description: This program causes various move, draw, and character commands to be passed to the Line Generator, so that execution times for the commands may be verified with the aid of an oscilloscope. The phases of the program are:

1. 19.8" Draw, Draw, Move (DMA)
2. 19.8" Draw, Draw, Dot (DMA)
3. 6300 10-point characters (Autorefresh)
4. 7000 4-point characters (Autorefresh)
5. 6300 10-point characters (DMA)
6. 7000 4-point characters (DMA)
7. 9.9" Draw (DMA)
10. 1/2" Draw (DMA)
11. 1/5" Draw (DMA)
12. Big X (DMA)

Program Description:

DMA Phases (1,2,5 through 12):

a. If first pass for this phase, reset the Picture System and announce the phase.

b. Load the data buffer with data for this phase. Data may consist of two buffers, one containing initializing data which is output once for each refresh, and a "repeatable" buffer to be output a specified number of times during each refresh. Initialize the phase counter.

c. Wait for real-time clock. Output the initial data buffer via DMA. Output the repeatable buffer via DMA as required.

d. Decrement and test the phase counter. If non-zero, go to step c, otherwise exit.

Autorefresh Phases (3,4)

a. If first pass for this phase, reset the Picture System and announce the phase.

b. Load the initial data buffer into Picture System memory. Output the repeatable buffer to Picture System memory as many times as required. Start autorefresh. Initialize the phase counter.

c. Wait for real-time clock.

d. Decrement and test the phase counter. If non-zero, then go to step c, otherwise exit.

Name: MAP Drawing Modes Test

Functional Description: This program visually demonstrates the operation of the MAP finite state machines FSM1 and FSM2. In all phases, the same sequence of 5 two-dimensional coordinates are processed by the MAP, using various combinations of FSM1 and FSM2:

PHASE	FSM1	FSM2
1	0 (M,D,M,D)	0 (SB,0,0,0)
2	1 (D,M,D,M)	0
3	2 (D,D,D,D)	0
4	3 (M,M,M,M)	0
5	0	3 (R,R,R,R)
6	1	3
7	2	3
10	3	3
11	0	2 (A,R,R,R)
12	1	2
13	2	2
14	3	2
15	0	4 (A,A,A,A)
16	1	4
17	2	4
20	3	4

(Note: M=Move, D=Draw, SB=Set Base, O=Origin Offset, A=Absolute, and R=Relative)

Program Description:

a. If not first pass for this phase, go to step d.

b. Announce the phase, and reset the Picture System.

c. Reset the MAP, enable MAP Active Output, and set MAP Active Output Address equal to the Line Generator Passive Input Port. Clear MAP Halt.

d. Load the data buffer with data to draw a reference grid, put the Line Generator in dot mode, a 2-d draw RSR command with FSM1 and FSM2 for this phase, and the following octal coordinates:

X	Y
-10000	0
-10000	-20000
10000	-20000
10000	0
30000	20000

e. Initialize the phase counter.

f. Wait for real-time clock.

g. Send the data buffer via DMA to the MAP Passive Input Port.

h. Decrement and test the phase counter. If non-zero, go to step f, otherwise exit.

Expected Visual Patterns: Figure RSD005-1 illustrates the expected pattern for each phase.

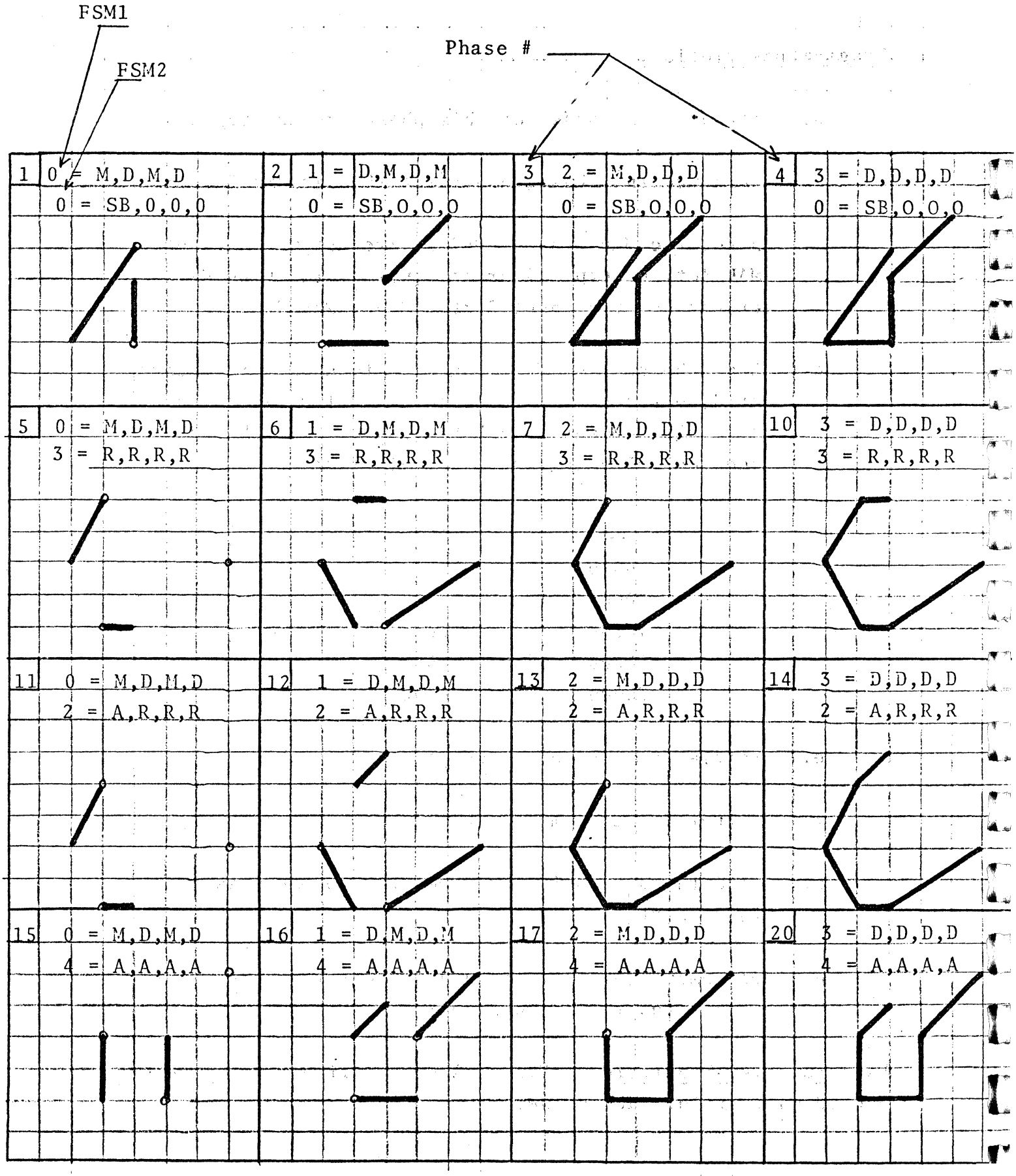


FIGURE RSD005-1

Identification: RSD006
Date: 11/3/78

Name: Refresh Rates Test

Functional Description: This program visually demonstrates various autorefresh rates:

PHASE 1:	24Hz
PHASE 2:	30Hz
PHASE 3:	40Hz
PHASE 4:	60Hz
PHASE 5:	120Hz

Program Description:

a. If not first pass for this phase, go to step d.

b. Announce the phase, reset the Picture System, and load the data buffer with a pattern of zig-zag lines.

c. Start autorefresh, using the rate for this phase.

d. Initialize the phase counter.

e. Wait for real-time clock.

f. Decrement and test the phase counter. If non-zero, go to step e, otherwise exit.

Identification: RSD007
Date: 11/3/78

Name: Line Generator Functions Test

Functional Description: This program visually demonstrates various Line Generator functions. Phases 1-10 use DMA, and phases 11-20 use Autorefresh. P100 determines the refresh rate, default 60Hz. The functions demonstrated are as follows:

PHASE 1/11:	Solid Lines
PHASE 2/12:	Long Dashes
PHASE 3/13:	Short Dashes
PHASE 4/14:	Long-short Dashes
PHASE 5/15:	Long-short-short Dashes
PHASE 6/16:	Blink
PHASE 7/17:	Dots
PHASE 10/20:	Raster

The raster phase is useful for detecting scope burns, and testing detectable line width and scope brightness.

Program Description:

a. If not first pass for this phase, go to step d.

b. Announce the phase, reset the Picture System, and initialize the refresh and real-time clock rate, according to P100. If P100 = 0, set the rate to 60Hz.

c. Load the data buffer with data for this phase.

d. Initialize the phase counter.

e. If phase 1-10 (DMA phases), go to step i.

f. Transfer the data buffer to Picture System memory via Direct I/O. Start autorefresh.

g. Wait for real-time clock.

h. Decrement and test the phase counter. If non-zero, go to step g, otherwise exit.

i. Wait for real-time clock.

j. Transfer the data buffer to the Line Generator Passive Input Port via DMA (do not wait).

k. Decrement and test the phase counter. If non-zero, go to step i, otherwise exit.

Identification: RSD008
Date: 11/3/78

Name: Scope Selection Test

Functional Description: This program verifies the scope selection function in the Line Generator. Phases 1-7 use DMA, and phases 11-17 use Autorefresh. P100 determines the refresh rate, default 60Hz. The phases are as follows:

PHASE 1/11:	Select all Scopes
PHASE 2/12:	Select Scope 0
PHASE 3/13:	Select Scope 1
PHASE 4/14:	Select Scope 2
PHASE 5/15:	Selcet Scope 3
PHASE 6/16:	Select Scope 4
PHASE 7/17:	Select Scope 5
PHASE 10:	Do Nothing

Program Description:

The data for each phase consists of Line Generator initialization including the appropriate scope selection, followed by a test pattern of concentric squares. The program steps are identical to those of RSD007.

Name: Character RAM Visual Test

Functional Description: This program is used for testing Character RAM and non-standard Character ROM, particularly when it is necessary to initialize the X and Y displacement registers. All phases use autorefresh. The phases are as follows:

Phase 1: Full Character Set, ROM, Display Control-Characters.

Phase 2: Full Standard Character Set, RAM.

Phase 3: 7000 4-point Fast Characters, RAM.

Phase 4: 6300 10-point Fast Characters, RAM.

Program Description:

a. If not first pass for this phase, got to step d.

b. Announce the phase, and reset the Picture System.

c. Load the data buffer with data to clear the x and y displacement registers (all phases) and load the Character RAM with the character definitions for the standard character ROM (Phases 2-4). Output the data buffer to the Line Generator Passive Input Port via Direct I/O.

d. Load Picture System memory with the data for this phase, via Direct I/O.

e. Start autorefresh, and initialize the phase counter.

f. Wait for real-time clock.

g. Decrement and test the phase counter. If non-zero, go to step f, otherwise exit.

Name: MAP Transformations Test (Version 1)

Functional Description: This test visually demonstrates the following MAP functions: clipping, perspective, viewport mapping, depth cueing, translation, rotation, scaling, and hit testing. It confirms that Autorefresh and MAP Active Input and Output are functioning. Visual patterns may be created to check Line Generator short vector adjustment. Either of two objects may be displayed, a three-dimensional "jack", or a two-dimensional grid composed of 2065_{10} short vectors. Either tablet or function switches may be used as the controlling input device. The grid pattern is useful in testing Picture System Memory, and Tablet dipswitch settings determine tablet resolution. This test does not use DMA or interrupts. The program consists of four phases:

Phase	Device	Object
1	Tablet	Jack
2	Function Switches	Jack
3	Tablet	Grid
4	Function Switches	Grid

Program Desciption: The following description applies to all four phases.

a. Reset the Picture System

b. 16K of Picture System Memory beginning at the address specified in P100 is divided into three areas as follows: one MAP Input Buffer, and two Refresh Buffers for double buffering. MAP Active Input is used to read the MAP Input Buffer, and MAP Active Output is used to write into

either refresh buffer. Autorefresh is used to display either refresh buffer. For the Multi-User Refresh Control, the last 200_8 words of the 16K area are used for Refresh Control Tables.

c. Initialize the MAP input buffer with data to (1) initialize the viewport, base, and matrix stack pointers, (2) initialize the transformation matrix, (3) initialize the Line Generator, (4) draw the viewport borders and menu, (5) draw the tablet cursor if applicable, (6) concatenate a window matrix, translation matrix, three rotation matrices, and one scale matrix, (7) draw the selected object, and (8) halt the MAP.

d. Initialize and enable MAP Active Input and MAP Active Output, and start the MAP.

e. Poll the selected input device. (Tablet or Function Switches). The available functions correspond to function switches or areas of the menu from left to right as follows:

(1)	Terminate Phase (HALT)
(2-4)	Unused
(5)	Enable/Disable Hit Testing (HIT TEST)
(6)	Select the next update rate equal to 1,2,4, or 8 refresh cycles per frame update (RATE 1/N)
(7)	Change the W scale factor (SCALE W)
(8)	Change the Z scale factor (SCALE Z)
(9)	Change the Y scale factor (SCALE Y)

(10) Change the X scale factor (SCALE X)

(11) Rotate about the Z axis (ROT Z)

(12) Rotate about the Y axis (ROT Y)

(13) Rotate about the X axis (ROT X)

(14) Translate in Z (TRAN Z)

(15) Translate in Y (TRAN Y)

(16) Translate in X (TRAN X)

Functions 7 through 16 each have a state equal to Ø through 3, and a changing transformation factor associated with them. For example each scale function has a value in the scale matrix as its transformation factor, and each rotation function has an angle of rotation in units such that 1024 equals one full revolution. For each function, a state of Ø or 2 means that the associated transformation factor is not changing, i.e. the function is inactive. A state of 1 means that the transformation factor is increasing by a fixed amount per frame update, and a state of 3 means that the factor is decreasing by the same amount. The scale and translation functions have upper and lower limits. When either limit is surpassed, the state of the function changes from an odd value to the next higher even value, which means that the function becomes inactive. For example, the X, Y, and Z scale factors have a lower limit of Ø. When the SCALE X factor equals Ø, the state of the SCALE X function changes from 3 to Ø, and the SCALE X factor ceases to change. If the SCALE X function is selected again by toggling switch 10 or touching down in the SCALE X area of the menu, the state of

the function becomes 1, and the SCALE X factor begins to increase. The polling operation determines which function has changed state, if any. If the input device is the tablet, the 195181 switches 3, 4 in location 42 determine tablet resolution as follows:

SW3	SW4	
0	0	100 units per inch
0	1	200 units per inch
1	0	400 units per inch
1	1	800 units per inch

The first two cases are in agreement with the PS-2 Graphics Software Package. This facility is provided as a means of verifying the switch settings.

f. If HALT was selected, Halt Autorefresh and exit.

g. Update the transformation factors for all active functions. Change state for any functions which have surpassed their limits. Update the transformation matrices in computer memory, and flag them for update in PS Memory. If Hit Testing is enabled, draw the Hit Window.

h. Wait for 1,2,4, or 8 real-time clock requests.

i. Wait for MAP done (PPDONE). Read the MAP Active Output Address for Refresh Limit. If Hit Testing is enabled, inhibit MAP Active Output, write the Hit Window matrix into MAP Input Buffer, and start the MAP, to perform a hit test.

j. Swap the refresh buffers and start autorefresh.

k. If Hit Testing is enabled, wait for PPDONE, and update the software HIT flag.

l. If the HIT flag is set, enable display of the "HIT" message, and clear the HIT flag.

m. Update the menu function states and cursor position in the MAP input buffer in PS Memory.

n. Update the transformation matrices in the MAP input buffer in PS Memory.

o. Go to step (d).

Identification: RSD011
Date: 10/19/78

Name: MAP Transformations Test (Version 2)

Functional Description: This test visually demonstrates the following MAP functions: Clipping, perspective, viewport mapping, depth cueing, translation, rotation, scaling, and hit testing. It confirms that Autorefresh and MAP Active Input and Output are functioning. Visual patterns may be created to check Line Generator short vector adjustment. Either of two objects may be displayed, a three-dimensional "jack", or a two-dimensional grid composed of 2065_{10} short vectors. Either light pen or 32 function switches may be used as the controlling input device. The grid pattern is useful in testing Picture System Memory. The Light Pen phases will operate only with Multi-User Refresh Control. This test does not use DMA or interrupts. The program consists of four phases:

Phase	Device	Object
1	Light Pen	Jack
2	32 Function Switches	Jack
3	Light Pen	Grid
4	32 Function Switches	Grid

Program Description: The following description applies to all four phases.

a. Reset the Picture System

b. 16K of Picture System Memory beginning at the address specified in P100 is divided into three areas as follows: one MAP Input Buffer, and two Refresh Buffers for double buffering. MAP Active Input is used to read the MAP Input Buffer, and MAP Active Output is used to write into

either refresh buffer. Autorefresh is used to display either refresh buffer. For the Multi-User Refresh Control, the last 200_8 words of the 16K area are used for Refresh Control Tables.

c. Initialize the MAP input buffer with data to (1) initialize the viewport, base, and matrix stack pointers, (2) initialize the transformation matrix, (3) initialize the Line Generator, (4) draw the viewport borders and menu, (5) draw the light pen tracking cross if applicable, (6) concatenate a window matrix, translation matrix, three rotation matrices, and one scale matrix, (7) draw the selected object, and (8) halt the MAP.

d. Initialize and enable MAP Active Input and MAP Active Output, and start the MAP.

e. Poll the selected input device. (Light Pen or 32 Function Switches). The available functions correspond to function switches or areas of the menu from left to right as follows:

(1)	Terminate Phase (HALT)
(2-4)	Unused
(5)	Enable/Disable Hit Testing (HIT TEST)
(6)	Select the next update rate equal to 1,2,4, or 8 refresh cycles per frame update (RATE 1/N)
(7)	Change the W scale factor (SCALE W)
(8)	Change the Z scale factor (SCALE Z)
(9)	Change the Y scale factor (SCALE Y)

(10) Change the X scale factor (SCALE X)

(11) Rotate about the Z axis (ROT Z)

(12) Rotate about the Y axis (ROT Y)

(13) Rotate about the X axis (ROT X)

(14) Translate in Z (TRAN Z)

(15) Translate in Y (TRAN Y)

(16) Translate in X (TRAN X)

Functions 7 through 16 each have a state equal to Ø through 3, and a changing transformation factor associated with them. For example each scale function has a value in the scale matrix as its transformation factor, and each rotation function has an angle of rotation in units such that 1024 equals one full revolution. For each function, a state of Ø or 2 means that the associated transformation factor is not changing, i.e. the function is inactive. A state of 1 means that the transformation factor is increasing by a fixed amount per frame update, and a state of 3 means that the factor is decreasing by the same amount. The scale and translation functions have upper and lower limits. When either limit is surpassed, the state of the function changes from an odd value to the next higher even value, which means that the function becomes inactive. For example, the X, Y, and Z scale factors have a lower limit of Ø. When the SCALE X factor equals Ø, the state of the SCALE X function changes from 3 to Ø, and the SCALE X factor ceases to change. If the SCALE X function is selected again by depressing switch 10 or touching down in the SCALE X area of the menu, the state of

the function becomes 1, and the SCALE X factor begins to increase. The polling operation determines which function has changed state, if any.

f. If HALT was selected, HALT Autorefresh and exit.

g. Update the transformation factors for all active functions. Change state for any functions which have surpassed their limits. Update the transformation matrices in computer memory, and flag them for update in PS Memory. If Hit Testing is enabled, draw the Hit Window.

h. Wait for 1,2,4, or 8 real-time clock requests.

i. Wait for MAP done (PPDONE). Read the MAP Active Output Address for Refresh Limit. If Hit Testing is enabled, inhibit MAP Active Output, write the Hit Window matrix into Input Buffer, and start the MAP, to perform a hit test.

j. Swap the refresh buffers and start autorefresh.

k. If Hit Testing is enabled, wait for PPDONE, and update the software HIT flag.

l. If the HIT flag is set, enable display of the "HIT" message, and clear the HIT flag.

m. Update the menu function states in the MAP input buffer in PS Memory. The tracking cross, if enabled, is positioned and drawn by the Multi-User Refresh Controller.

n. Update the transformation matrices in the MAP input buffer PS Memory.

o. Go to step (d).

Identification: RSD012
Date: 10/22/78

MP101 = XXXX
E

Name: Peripherals Visual Test

Functional Description: This program may be used to visually verify the operation of all PS-2 peripherals. The contents of all data and status registers for the peripherals selected by the user are sampled and displayed as octal numbers in a refresh buffer. The program consists of two phases, both non-terminating. Phase 1 loads the refresh buffer into PS Memory, and uses Autorefresh. The refresh buffer is updated via the Direct I/O path, and no DMA's are used. Phase 2 uses a refresh buffer in host computer memory, which is transmitted to the Line Generator passive input port via DMA for every refresh cycle.

Operation of the program, aside from device address designation, is controlled by four locations in the Picture System Table (PSTB) as follows:

P76	Device Acknowledge Countdown +1, default = 31 octal (see steps m and o of program description).
P77	Mask for A/D Data, default = 177700.
P100	Device Selection, default = 307.
P101	Number of A/D Channels, default = 10 octal.

174374
20

For each of these locations, a value of zero results in use of the default value.

Following are the devices which may be selected, the bit in P100 which enables the device, and the PSTB location containing the base address of the device.

P100 Bit	Device	Base Address
0	Tablet	P33
1	Function Switches	P41
2	A/D Interface	P47
3	Keyboard	P36
4	32 Function Switches	P55
5	Light Pen	P44
6	Device Request	P34
7	System Request	P20
8	Tablet 2	P60
9	Function Switches 2	P66
10	A/D Interface 2	P74
11	Keyboard 2	P63
12	32 Function Switches 2	P102
13	Light Pen 2	P71
14	Device Request 2	P61
15	System Request 2	P25

Note that the default value for P100, 307_8, selects Tablet, Function Switches, A/D Interface, Device Request, and System Request.

Program Description

a. Reset the Picture System.

b. For each selected device, load data into the display buffer in computer memory.

c. If phase 1, transfer the display buffer to PS Memory, and start autorefresh.

d. For each selected device, perform steps h through k to update the display buffer in PS Memory for phase 1, or computer memory for phase 2.

e. Wait for real-time clock request.

f. If phase 2, then transmit the display buffer to the Line Generator passive input port via DMA.

g. Go to step (d).

h. Steps i through k describe the procedure for updating the display file for each item. Each selected device has one or more display items associated with it. For example, the Tablet has four display items: a header, the tablet status register, the X data register, and Y data register. The initialization and updating of each item is controlled by 16 words constituting an entry in the Item Table.

i. If enabled, read one word of data via DIO, from the PS-2 SCB address associated with the item. AND the received data with a specified mask, and compare with the old value. If no change, go to step k.

j. If octal/ascii conversion is enabled, convert the received data to an ascii string and write the string into a specified position in the display file (Phase 1, PS Memory; Phase 2, computer memory).

k. If a special subroutine is specified, call the subroutine, then return. Steps 1 through o describe the special subroutines.

1. A/D Interface Subroutine: if not first time called, then return. If the present channel number is greater than P101 (default 8), then disable all future calls

to this item. If P77 is non-zero, save it as the read-mask for this item. Return.

m. Device or System Request Subroutine: if first time, and P76 is non-zero, save it as writeback count. Call the writeback subroutine (see step o below).

n. Keyboard Subroutine: if the received value is within the range of displayable characters, display the character (in addition to the octal character code which is displayed in any case.)

o. Writeback Subroutine: if the writeback counter for this item is non-zero, decrement it and exit. Otherwise, re-initialize the counter and write the received value into the specified writeback SCB address. This subroutine is used to write the function switches back to the function lights, and to acknowledge device and system interrupt requests. The purpose of the default writeback count of 30 octal is to allow device requests to stay high in the display file long enough to be easily observed. In some cases, however, this causes the tablet to update its data registers slowly. For maximum tablet update rate, P76 should be set equal to 1, as 0 implies the default value.

CHAPTER SIX

6.1 INTRODUCTION TO QSDDT

QSDDT is a general purpose PS-2 diagnostic tool written in the MIXIT programming language. QSDDT provides commands to examine and modify registers or memory locations in 12_{10} addressing modes as follows:

- M: PS-2 Memory/SCB (Ø to 177777)
- B: Host Computer Memory Buffer (Ø to 177)
- I: Host Computer Interface Registers (Ø to 4)
- D: Picture Processor DOIT Register (Subfields Ø to 5)
- C: Picture Processor Control Store (Ø to 377)
- P: Picture System Device Table (PSTB)
- H: Host Computer Table (HSTB)
- A: Character Memory (Ø-1777,2000-3777)
- E: Character Generator Coefficient Memory (Ø-77, 200-377)
- L: Line Generator/Character Generator State (Ø-37)
- Q: Inter-character and Inter-line Spacing Registers (10-13)
- F: Autorefresh Parameters (Ø-2)

In addition to commands which examine and modify locations, QSDDT supports other commands including the following: (a) reset the Picture System, (b) wait for DMA ready, (c) wait for Real-Time Clock, (d) save, unsave or verify the MAP DOIT Register, (e) specify octal or decimal input, (f) transfer data from one location to another, (g) execute a command line repeatedly, and (h) compare a received value with an expected value.

All command strings are terminated with a carriage return. Processing of a command string consists of an interpretation phase followed by an execution phase. Interpretation of a repeated command string is performed once only, allowing relatively high speed execution.

6.1.1 Introductory Examples

It is not necessary to know all commands in order to use QSDDT. A few of the simplest commands are also the most useful commands. Other commands may be learned as need arises. For reference, section 6.17 contains a summary of all QSDDT commands, with section numbers for more detailed information. The following examples are enough to make QSDDT useful in many cases:

R	Reset Picture System
MØ	Display contents of PS Memory location Ø
MØ=200,176000	Deposit 200 (octal) in PSMEM(Ø) and 176000 in PSMEM(1)
FØ=0,100,17;G	Set refresh start address =0(FØ) refresh limit=100 (F1) clock rate=120(F2) and start autorefresh (G)

K	Kill autorefresh
MØ:17	Display PSMEM(Ø) through PSMEM(17)
MØ M=125252,52525,$1000	Load PSMEM(Ø) through PSMEM(1777) with checkerboard pattern
MØ ?125252,?52525,$1000	Verify the checkerboard pattern, and report the first error
X	Exit QSDDT

6.2 MEMORY COMMANDS

Following are the commands which open, examine, and modify registers (including memory locations) in the various addressing modes. Opening a location does not necessarily entail reading the location.

In the following discussion, "X" is an address mode symbol (M,B,I,D,C, etc), and "n" is a numeric value. "[n]" indicates a numeric specification which may be omitted.

The Memory Commands are as follows:

Xn Set addressing mode = X, and open location n. Example: MØ

=n Deposit n in the currently open location. Example: MØ=200

,[n] Open the location following the current location. If n is specified, deposit n in the newly opened location. Example: MØ=Ø,,1 modifies MØ and M2.

'[n] Open the location preceding the currently open location. If n is specified, deposit n. Example: M100=Ø'1 modifies M100, then M77.

. Display the contents of the currently open location. Example: M177760=1. modifies and then reads the same location back.

<CR> If the last opened location has not been modified, then display its contents. Example: MØ<CR>.

/ Read but do not display the currently open location. Example: CØ/ loads MAP Control Store Ø into the MAP DOIT register.

:n Display the contents of the block of locations beginning with the currently open location and ending with location n. Example: MØ:17.

> Transfer the last read data (as modified by * or N command) to the next opened location. Example: MØ>1 copies MØ into M1.

n According to the context established by the other commands in this section, n may be interpreted as the address of a location to be opened, or a value to be deposited. Either context may be forced by preceding n with = to deposit, or one of the address mode commands (M,B,I,D,C, etc.) to open. At the beginning of a line or following a semi-colon, n is interpreted as an address, in the current addressing mode. Example: MØ=2. Ø is an address, 2 is a value.

6.3 ADDRESSING MODES

The last issued address mode command defines the current domain of the memory commands which read, display, and modify registers or memory locations. Unless otherwise specified the addressable registers contain 16 bits. MAP Control Store memory locations consist of 96 bits which correspond to the six accessible 16 bit subfields of the DOIT register. The addressing modes are as follows:

6.3.1 (MØ:177777) PS-2 MEMORY/SYSTEM CONTROL BLOCK: In this mode all Picture System Memory locations, and registers of the PS-2 SCB, are accessible. The range of legal addresses is Ø through 177777, although many systems include less than the full memory configuration.

6.3.2 (BØ:177) HOST COMPUTER MEMORY BUFFER: The primary purposes of this mode are to facilitate DMA diagnosis, and to provide capability for refresh from a host computer memory buffer. When starting up, QSDDT reports the absolute address of the host buffer (DBUF = XXXXXX). Typically, the operator would deposit data in this buffer, then deposit the reported DBUF address in I3 (DMABA-PDP-11 only) in preparation for a DMA transfer. The reported DBUF address will be incorrect on a system using memory management.

Locations in the host buffer are addressed for purposes of examination and modification with a word index ranging from Ø to 177. To use a DMA start address other than at the beginning of the host buffer, the user must compute the absolute address within the buffer. In the case of the PDP-11, the absolute address is computed as DBUF + (2 x index).

6.3.3 (IØ:77) HOST COMPUTER INTERFACE REGISTERS: This mode addresses the registers on the host computer side of the PS-2 interface (PSBUS) as follows (PDP-11 only):

0 = PSDATA:	Direct I/O Data Register, default address = 767660 (195131 card)
1 = DIOPSA:	Direct I/O Picture System Address, default address = 767662 (195105 card)

2 = DMAWC:	DMA word count (two's complement), default address = 767664 (195131 card)
3 = DMABA:	DMA Host Buffer Address, default address = 767666 (195131 card)
4 = IOST:	I/O Status Register, default address = 767670 (195131 card)

For a non-standard UNIBUS configuration, the base address of this block of registers (default 767660) is modifiable as location H1 (see addressing mode "H" below).

Uses for "I" mode include diagnosis of the interface and general access to the I/O Status Register.

The range of legal addresses in I mode is Ø through 77, but the significance of each address may vary with different computers, and no address above 20 has been used on any computer at the present time. Documentation for "READ" and "WRITE" subroutines for each computer will detail the significance of each address in mode "I".

6.3.4 (DØ:5) PICTURE PROCESSOR DOIT REGISTER: The hardware DOIT register consists of eight virtual segments Ø through 7. The first 6 of these segments contain Picture Processor (MAP) control signals and represent the current state of

the MAP. Segments 6 and 7 of the DOIT do not represent control data, but are accessed to write the contents of the DOIT into the Control Store, and load the Control Store output into the DOIT respectively. The DOIT register is implemented on the 195115-100 and -101 cards. Systems containing the Writable Control Store have 195124 cards in lieu of 195115's.

QSDDT recognizes only segments Ø through 5 as legal DOIT addresses. Segments 6 and 7 are implicitly utilized in Control Store Addressing Mode.

6.3.5 (CØ:377) PICTURE PROCESSOR CONTROL STORE: This addressing mode is somewhat unique due to the fact that addressable locations contain 96 bits. Opening and displaying a location causes the contents of that location to be loaded into the DOIT register, which is thereafter displayed in six segments. Attempts to deposit data into Control Store locations will have no effect, of course, if the system does not contain the Writable Control Store option. Otherwise, the current contents of the DOIT will be written into the open Control Store location.

N.F.G

Modifying the MAP Writable Control Store:

Commands to modify a location (= , ') are effective only if followed by a numeric specification. Therefore, in order

to write the contents of the DOIT into Writable Control Store, a dummy numeric argument must be supplied with these commands. Example: C100=Ø causes the current contents of the DOIT (rather than the dummy value Ø) to be written into MAP Control Store location 100, provided the Picture System contains Writable Control Store.

6.3.6 (P2:64) PICTURE SYSTEM DEVICE TABLE (PSTB): This mode is used to establish non-standard addresses in the PS-2 System Control Block. The addresses reside in a software table called PSTB, the same table which is accessible by means of the "Modify" command in standard PS-2 Diagnostics (cf. Chapter 5).

6.3.7 (H1:13) HOST COMPUTER TABLE (HSTB): This mode provides access to HSTB, a table which defines the UNIBUS base address of the PS-2 device register (IOST, etc.) and the PS-2 Interrupt Vector Base. This is also a table which is accessible through the "Modify" command in standard PS-2 Diagnostics (cf. Chapter 5). Ordinarily, no modification of values in this table is necessary, unless multiple picture systems are interfaced to one processor.

6.3.8 (AØ:3777) CHARACTER MEMORY: This mode provides read-only access to the Character ROM (Ø-1777) and read/write access to the Character RAM (2000-3777) located on the 195222 card. Data is treated as 12_{10} bits right-justified within a 16_{10} bit software word. For the formats of Character Generator

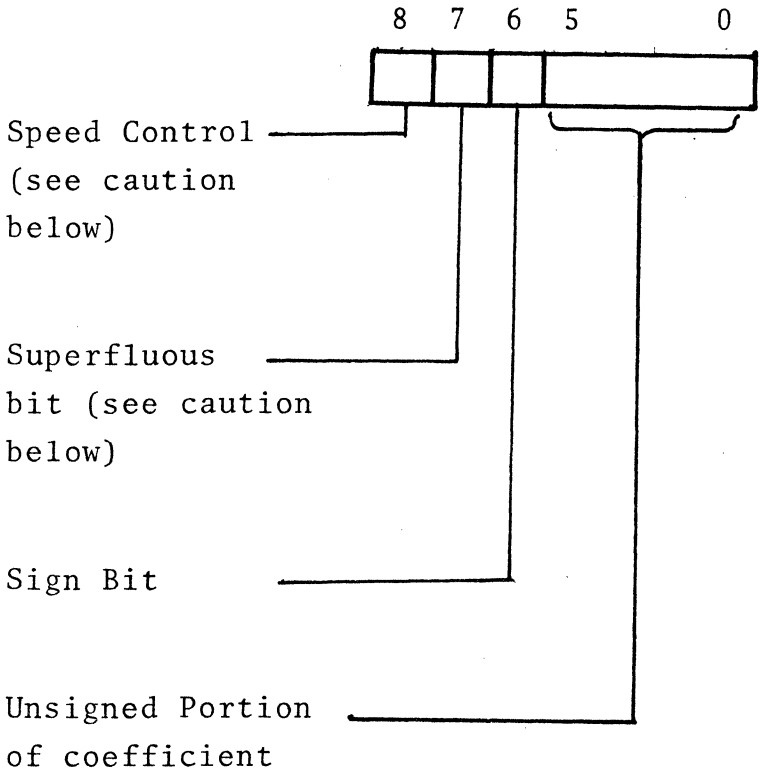
commands, see Section 2.4.4.3 of the PS-2 Reference Manual.

Caution: It is possible to cause scope burns by modifying Character Generator commands.

6.3.9 (EØ:377) CHARACTER GENERATOR COEFFICIENT MEMORY: (Located on the 195221 card) according to PS-2 Reference Manual Sections 2.4.4.4, 2.4.4.5.b, and 2.4.5 "Reading the Coefficient Memories", the address of a coefficient memory element consists of a value in the range Ø-17 (read only) or 40-77 (read/write), followed by a suffix A,B,C, or D. QSDDT treats coefficient memory addresses as having a range of Ø-77 (read only) and 200-377 (read/write). In the QSDDT addressing scheme, the two least significant bits of the address designate A,B,C, or D. Correspondence to the Reference Manual addressing scheme is as follows:

Reference Manual	QSDDT	
ØA	0	First read-only location
ØB	1	
ØC	2	
ØD	3	
⋮	⋮	
17C	76	
17D	77	Last read-only location
40A	200	First read/write location
⋮	⋮	
77D	377	Last read/write location

The format for coefficient Memory data is as follows: (see Reference Manual 2.4.4.4).



Caution--Speed Control: (a) Speed Control bits always read back as Ø, even though 1 way be written in. (b) The Speed Control bits of corresponding A/C or B/D elements should always be equal.

Caution--Superfluous Sign Bit: Bit 7 should always be equal to bit 6.

Caution--Scope Burns: It is possible to cause scope burns by modifying the Coefficient RAM.

6.3.10 (LØ:37) LINE GENERATOR/CHARACTER GENERATOR STATES: This data is read only. Addresses Ø-17 correspond to PGXBUS modes Ø-17. Addresses 20-37 correspond to PGYBUS modes Ø-17. See Section 2.4.5 of the PS-2 Reference Manual.

6.3.11 (Q10:13) INTER-CHARACTER AND INTER-LINE SPACING REGISTERS: This mode provides write only access in the range 10-13 as follows (see Reference Manual 2.4.4.5.c):

10: X 12-bit integer part
11: Y 12-bit integer part
12: X 12-bit fractional part
13: Y 12-bit fractional part

6.3.12 (FØ:2) AUTOREFRESH PARAMETERS: This address mode facilitates autorefresh control independent of the type of refresh controller (single-user or multi-user) available in the Picture System.

FØ: Refresh Start Address
F1: Refresh Limit
F2: Clock Rate (17=120 HZ, 16=60 HZ, etc.)

For further details, see Section 6.5, Autorefresh Control. Note that Autorefresh Parameters do not take effect until a "G" (Start Autorefresh) command is issued, even if autorefresh is already running. Also, in the case of a Multi-user Refresh Controller, autorefresh will not operate unless an "R" command

(PS Reset) is issued before the first "G" command.

NOTE: FØ and F1 should always be assigned even values. F1 should be set one greater than the last Picture System Memory location to be included in the refresh.

6.4 REPEATED EXECUTION ($ OR $N)

Both definite and indefinite repetition will terminate in the event of one of the following errors: (See section 6.15)

ADDRESS ERROR
READ ERROR
WRITE ERROR
COMPARISON ERROR

In the event of a SYNTAX ERROR or EXECUTION BUFFER FULL, no attempt will be made to execute the command string even once.

6.4.1 INDEFINITE REPETITION ($): The dollar-sign followed by no number or a negative number causes "non-terminating" execution of a command string (see exception above). At the end of a command string (e.g. M,=$) the string up to the $ is interpreted and executed. If a command string consists of "$" only, then the last preceding input command string will be re-executed, indefinitely.

6.4.2 DEFINITE REPETITION ($n): The dollar-sign followed by a positive number causes the command string up to the "$" to be re-executed the specified number of times. If a command string consists of $n (e.g. $20) only, then the last preceding input command string (up to but not including "$" character, if any) will be re-executed n times.

6.5 AUTOREFRESH CONTROL (G,K)

The "G" command causes autorefresh to be initiated according to autorefresh parameters FØ through F2 (see 6.3.12), and "K" causes autorefresh to halt. The "R" command (PS Reset) will also halt autorefresh.

Autorefresh parameters for a display buffer already in PS Memory may be determined as follows, prior to issuing a "G" command:

Refresh Start Address: With a Single-user Refresh Controller, examine M177735. With a Multi-user Refresh Controller, and a display buffer generated by a diagnostic, examine M37654. For such display buffers, the Refresh Start Address will normally be zero.

Refresh Limit: With a Single-user Refresh Controller, examine location M177736. With a Multi-user Refresh Controller, and a display buffer generated by a diagnostic, examine M37655. The contents of this location is two greater than the F1 parameter.

Clock Rate: The clock rate takes effect only upon occurrence of a "G" command. The same rate is used for both autorefresh and real-time clock, as follows:

F2 = 17	120 HZ
16	60 HZ
15	40 HZ
14	30 HZ
⋮	
F2 = Ø	previous rate is used

Special Considerations Concerning the Multi-user Refresh Controller: An "R" command (PS Reset) must be issued before the first "G" command in order for autorefresh to operate.

The "G" command causes the contents of the PS Memory location addressed by F1 to be saved, and a Refresh Controller halt command (40202) to be stored in that location. When F1 is changed, the next "G" command restores the previous limit location, saves the contents of the location now addressed by F1, and writes the halt command into the new limit location. FØ and F1 should always contain even values.

6.6 TEST FOR EXPECTED VALUE (?)

This command makes it possible to write simple memory tests in QSDDT (see section 6.1.1). A numeric value must always follow the question-mark, and represents the expected contents of the last opened location. If the expected value

does not equal the received value, execution of a command-line repetition ($) is terminated, and a message is printed as follows:

```
COMPARISON ERROR; EXPECTED=NNNNNN RECEIVED XN=NNNNNN
```

6.7 INPUT RADIX CONTROL (O,Z)

By default, all QSDDT input is octal. Execution of the commands "Z" and "O" set the input radix, however, to decimal and octal respectively. The effect of these commands is not limited to one command string, but persists until the complementary command is executed.

NOTE: All QSDDT output is octal, irrespective of the input radix. Thus, QSDDT facilitates decimal-to-octal conversion, but not the converse.

6.8 NUMERIC SPECIFICATION (-,")

A number may be specified with 1 to 6 characters as follows:

The first character may be a digit, a minus sign, or a double-quote character. A minus sign specifies a two's complement value (e.g. -1 = 177777). A double-quote character specifies a sign - extended value (e.g. "37 = 177737).

The input radix is octal by default, or depends upon the last executed radix command (O for octal or Z for decimal).

6.9 SYNCHRONIZATION COMMANDS (T,W)

The "T" command causes QSDDT to wait for a real-time clock request. The "W" command causes QSDDT to wait for DMA READY. These commands may be useful during repeated exectution of a command string, or to confirm operation of the real-time clock.

6.10 DOIT COMMANDS (S,U,V)

The "S" command causes the current contents of the MAP DOIT register to be saved in a software block of 6 words. The "U" command restores the contents of the software block to the DOIT register. These commands may be useful when single-stepping the MAP or modifying the MAP Writable Control Store. The "V" command causes the current contents of the DOIT to be displayed as six values on a single line. It is equivalent to "BØ:5" except for the display format.

6.11 QSDDT TERMINATION (X)

This command causes QSDDT to terminate and return control to the operating system.

6.12 PICTURE SYSTEM RESET (R)

The "R" command causes a Picture System Reset to be issued.

6.13 NEGATE AND COMPLEMENT (N,*)

These commands cause the following actions: (a) The contents of the currently open location is read into a software location, and (b) the contents of the software location is converted to the complement in the case of "*" or the two's complement in the case of "N". Note that these commands do not modify the contents of the currently open location, although the currently open location can be modified by a command such as "MØ*>MØ".

6.14 END OF STATEMENT (;)

The semi-colon is used to separate command statements within one line of input.

6.15 QSDDT ERROR MESSAGES

SYNTAX ERROR: Following this message, the input command string will be retyped up to the point at which the syntax error was detected. Note: Any spaces in a QSDDT command string will result in a syntax error.

ADDRESS ERROR: Following this message, the illegal address will be printed. This message results from violation of the range of legal addresses in the current mode.

WRITE ERROR: Following this message, the offending address will be printed. This error results from the attempt to write into a read-only location.

READ ERROR: Following this message, the offending address will be printed. This error results from the attempt to read a write-only location (address mode "Q"). If the command string entails displaying the contents of the location, invalid display will occur following the error message.

EXECUTION BUFFER FULL: The execution buffer is loaded with command indices and numbers during interpretation of the input command string. It is possible, though abnormal, for the execution buffer to overflow, in which case a shorter command string must be typed.

6.16 EXAMPLES

6.16.1 General Examples: Following are some general examples of QSDDT commands. The underscored lines are the input command strings. The other lines are printed by QSDDT.

```
>RUN QSDDT
DBUF = xxxxx                    address of BØ
```

```
%
M''37                          examine RF Status Register
                               (Single-user Refresh Con-
                               troller)
M177737 = 100000
%
ZBØ=100.                       set decimal input radix,
                               deposit decimal 100 in
                               BØ, display BØ
BØ = 144
%
O                              restore octal input radix
%
BØ*>B1.                        deposit BØ complement in
                               B1 and display
B1 = 177633
%
BØ:2                           display BØ through B2
BØ = 144 177633 4757
%
=Ø.                            modify and display B2
B2 = Ø
%
PØ                             attempt to open PSTB[Ø]
ADDRESS ERROR PØ               illegal address
%
,,                             advance to PSTB[2]
P2 = 177744
%
```

6.16.2 Line Generator Test Pattern: The following sequence of commands will cause a test pattern to be displayed by DMA to the line generator.

```
>RUN QSDDT
DBUF = xxxxx
%
R                                   reset the Picture System
"47                                 examine DMAPSA
M177747 = 177777                    DMA was directed to MAP
                                    passive input port
%
="5.                                redirect to Picture Gen-
                                    erator passive input port
M177747 = 177775
%
BØ=300,176000,133776,3776           load host buffer with
                                    data to draw a big "T"
,170000,3776,170000,4002
%
,134002,3776,170000,3776
%
,170000,4002.                       display last location to
                                    check word count
B15 = 4002
%
WI2=-16,xxxxx,1$10000               (PDP-11 only) this line
                                    waits for DMA ready, loads
                                    DMAWC with negative word
```

count, DMABA with the buffer address which was announced when QSDDT started up, sets "GO" in IOST, then goes back to wait for DMA ready and repeat 10000 times. A big "T" should now be visible on the scope

6.16.3 MAP Control Store Example: The following example examines and then modifies the contents of MAP Control Store location 4:

```
%
C4                                  read C4 into DOIT
C4 = 2367 57777 177777 177673 1777375 177777
%
DØ = 1,2,3,4,5,6/;V                 load the DOIT with new
                                    data and verify
DOIT = 1 2 3 4 5 6
%
C4 = Ø                              dummy write operation
                                    causes DOIT to be written
                                    into C4
%
```

6.16.4 Dump MAP Internal Registers into PSMEM

```
*RU QSDDT
DBUF=XXXXX
```

```
%
R                                   reset
%
M27=Ø
%
M"53=1=40,Ø                         reset MAP, then set MAO,
                                    then MMSR=Ø
%
M"50=377,Ø                          MAOL, MAOA
%
M-1=12001=Ø                         MPIP:  RSR Store(377,Ø)
%
M27
M27=157                             MAP stack ptr should equal
                                    117,137 or 157 etc.
%
MØ:3Ø                               this will display the
                                    first 30 locations, for
                                    example
MØ = XXX XXXX XX XXX
M4 = XX XXXXX XXX XX
etc.
%
```

6.16.5 Dump MAP Internal Registers (by DMA) into Host Computer Buffer.

```
*RU QSDDT
%
```

R	reset
%	
B27=Ø	
%	
M"53=0,"Ø	MAOL=Ø,MAOA=17777Ø; Point MAP output at DMA Passive Input Port (DMAPIP)
12=-200,XXXXX	(PDP-11 only) DMAWC (200 is size of DBUF); DMABA= DBUF
%	
I4=14=1	(PDP-11 only) IOST: DMAIN, PASSIVE, then set GO
%	
I4	
I4=100014	(PDP-11 only) DMA not ready
%	
M-1=12200=0	RSR Store (200,0)
%	
I4	
I4=140214	(PDP-11 only) DMA is now ready
%	
B27	
B27=137	MAP stack ptr should equal 117,137 or 157 etc.
%	

6.16.6 Modification of Character RAM (Mode A)

```
*RU RSD009
RSD009.S02
CHARACTER GENERATOR VISUAL TEST
%
D
DO WHICH PHASE(S)?
2
%
X
RUNNING
PH 2:  E&S STANDARD CHARACTER SET, RAM
PHASE 2 DONE
TEST COMPLETE
*RU QSDDT
DBUF = xxxxx
%
A3764:3770                          Display the stroke defini-
                                    tions for fast lower-case
                                    "u"
A3764 = 2004 2414 2500 2404
A3770 = 1140
%
3764=Ø                              Still in "A" mode, change
                                    the first move to a no-op,
                                    observe the fast "u" drop
                                    below its line on the
                                    screen
```

%

,Ø — Change the next stroke to a no-op, observe the effect. Similarly, 3766 and 3767 may be changed. NOTE: 3770 is a halt command and should not be changed to a no-op.

For moves and draws, bits 7-4 = delta x and bits 3-Ø = delta y. Consult reference manual Section 2.4.4.3 for more detailed information.

6.16.7 Modification of Coefficient RAM (Mode E)

Execute QSDØ27 Phase 15 in similar fashion to the preceding example.

```
*RU QSDDT
DBUF = xxxxx
%
E200:203
E200 = 42 0 0 42
%
```

E200=40 — Note the effect of this example on a line in the top-left quadrant

,40 — Text blows way out of proportion because a speed bit in location 201, which always reads

```
                                        as Ø, got changed from
                                        1 to Ø
=440.                                   Restore the speed bit
E201 = 40
%
E202=1Ø
%
,420                                    This changes 203.  Speed
                                        bits of 203 and 201 are
                                        equal.  200-203 = 40A-D,
                                        which is the first matrix
                                        in the coefficient RAM.
E203 = 20
%
```

Other RAM locations in the range 200-377 affect other characters on the screen.

6.16.8 Modification of Spacing Registers (Mode Q)

Execute QSDØ27 phase 12 in order to load the Character RAM (A2000 +) with the "box" definition which adds displacement after drawing the box character.

```
*RU QSDDT
DBUF = xxxxx
%
M"35.,                                  (or M37654.,) Read Refresh
                                        Start Address and Refresh
                                        Limit
```

R	Reset Picture System
MØ=300,176000,130000,170000	LG Reset, Move
%	
,37003,1006,0,0	(PDP-11 only) Load font: Ø is the box character. This is the end of the refresh buffer. For machines with first byte on the left side of each word: 1476,3002,0,0.
M207=Ø	
FØ=Ø,1Ø,17;G	Refresh Start Address, Refresh Limit, Clock Rate, Start Autorefresh. Four boxes should now appear on the screen.
%	
Q10=10	X integer part of spacing registers; note the screen
%	
=20	Increase delta X
%	
=200	
%	
,100	Y integer part
%	
=200	
%	

,400	X fractional part has negligible effect, as does Y fractional part.
,200	

6.16.9 Searching a Refresh Buffer

Suppose that a "glitch" appears on the screen and it is desirable to determine its exact location in the refresh buffer. This can be determined by doing "binary search" with the refresh limit. Run the phase of the diagnostic which produces the glitch (an autorefresh phase) and then run QSDDT.

%	
R	Reset-necessary only for Multi-user Refresh Controller.
%	
M177736	
M 177736 = 1000	Determine the refresh limit. For our example, assume 1000. For a Multi-user Refresh Controller, examine M37655, and subtract 2.
F1=400G	Default values are FØ=0 and F2=17 (120 HZ). Cut

the refresh limit in half, and restart autorefresh. If the glitch disappears, try F1=600G, or if it is still in the buffer, F1-200G. Continue in this way to narrow down the location of the glitch.

6.16.10 DMA to Line Generator (NORD-10)

Reset, then initialize M"47 and BØ through B15 as in example 6.16.2.

```
WI5=16;I3=XXXXX,4004$10000
```

This line waits for DMA ready, loads DMA word count, loads DMA start address, and loads DMA control, setting "write" and "activate" bits. These actions are repeated 10000 times.

6.16.11 DMA to Line Generator (Interdata 8-32)

Before running QSDDT, display the System Memory Partitions ("D<space>M<CR>"). Convert the hexadecimal base address of the ".BG" partition to octal (for example, hex 10400 = octal 202000). Now run QSDDT, reset the Picture System, and initialize M"47 and BØ through B15 as in example 6.16.2. The DMA start address must be computed by adding the .BG base address to the reported DBUF address (for example, 202000+57710=261710). The least significant 16 bits are loaded into I10 (I12 for DMA end address), and the most significant 4 bits into I11 (I13 for DMA end address).

I11=1,61742,1	Initialize extended part of DMA start address, DMA end address, and extended DMA end address.
WI1Ø=61710;I6=0=1$10000	This line waits for DMA ready, loads DMA start address, clears DMA control, then sets "go" in DMA control. These actions are repeated 10000 times.

6.17 SUMMARY OF QSDDT COMMANDS

CHARACTER	MEANING	SECTION
$	Repeat command string indefinitely	6.4.2
$n	Repeat command string n times	6.4.2
'[n]	Open preceding location and deposit n	6.2
*	Read and complement	6.12
,[n]	Open succeeding location and deposit n	6.2
.	Read and display contents of currently open location	6.2
/	Read the currently open location	6.2
:n	Read and display from current location to location n	6.2
;	End of command statement	6.13
=[n]	Deposit n in the current location	6.2
>Xn	Deposit the current value in location n	6.2
?	Compare received and expected values	6.6
A	Character Memory Address Mode	6.3.8
B	Host Buffer Address Mode	6.3.2
C	Control Store Address Mode	6.3.5
D	DOIT Address Mode	6.3.4
E	Coefficient Memory Address Mode	6.3.9
F	Autorefresh Parameters	6.3.12
G	Start Autorefresh (GO)	6.5
H	Host Table (HSTB) Address Mode	6.3.7
I	Interface Register Address Mode PSDATA, DIOPSA, DMAWC, DMABA, IOST (Ø through 4)	6.3.3
K	Halt Autorefresh (KILL)	6.5
L	Line Generator/Character Generator State Mode	6.3.10
M	PS Memory/SCB Address Mode	6.3.1
N	Read and negate	6.12
O	Octal Input	6.5
P	Picture System Device Table Address Mode (PSTB)	6.3.6
Q	Inter-Character/Inter-Line Spacing Address Mode	6.3.11
R	Reset Picture System	6.11
S	Save DOIT	6.8
T	Wait for Real-Time Clock	6.7
U	Unsave DOIT	6.8
V	Verify DOIT	6.8
W	Wait for DMA ready	6.7
X	Exit QSDDT	6.9
Z	Decimal Input	6.5

APPENDICES

APPENDIX A
E&S DIAGNOSTIC PACKAGE INDEX

A.1 E&S DISTRIBUTION NAMING CONVENTION

The E&S Diagnostic Package occurs in various distributions tailored to the configuration of various target systems. The name of a distribution takes the following form:

ES-PS[x][z] DSrrvv-dd[ddd][-yy]

The components of this name identify features of the distribtuion as follows:

PS[x][z]: x and z are null if the distribution includes PS1 diagnostics only.

x=2 and z is null if the distribution includes PS2 diagnostics only.

x=1 and z=2 if the distribution includes diagnostics for both PS1 and PS2.

DSrrvv: rr=release number, and vv=version number.

dd[ddd]: If this field consists of the two characters "PT" then the distribution medium is paper tape, and the software operation base is the paper tape loader.

If this field consists of five characters, then the first two characters designate the distribution medium (RK for RK05, DT for DECtape, or MT for Magtape), and the last

three characters designate the operating system (ESD or RSX).

[-yy]: If this field is omitted, then the distribution medium is also the system device. Otherwise, this field indicates the target system device.

A.2 DISTRIBUTIONS OF THE E&S DIAGNOSTIC PACKAGE

The currently available or anticipated distributions of the E&S Diagnostic Package are (November 1978):

(a) PS1 only:

ES-PS-DS0202-RKESD (RK05)
ES-PS-DS0202-DTESD (DECtape)

(b) Both PS1 and PS2:

ES-PS12-DS0307-RKESD (RK05)
ES-PS12-DS0307-MTESD-RK (Magtape/RK05)

(c) PS2 only (note---RSX does not support PS1 Diagnostics):

ES-PS2-DS0307-DTESD (DECtape)
ES-PS2-DS0307-RKRSX (RK05)
ES-PS2-DS0307-MTRSX-RK (Magtape/RK05)
ES-PS2-DS0304-MTRSX-DB (Magtape/RP04/05/06)
ES-PS2-DS0307-MTRSX-TSK (Magtape/Tasks created with FLX in /DO format)
[220,15] for Unmapped RSX-11M
[220,14] for Mapped RSX-11M
ES-PS2-DS0307-DXRT11 (RX01 Floppy, RT-11)

A.3 E&S DIAGNOSTIC NAMING CONVENTION

All E&S PICTURE SYSTEM Diagnostics and Programs are named, by convention, as follows:

PSTxxx.Yzz (For example PSD001.P01 or QSD002. S01)

Where:

PS is the Diagnostic Class designation:

PS for PICTURE SYSTEM 1 Diagnostic, QS for PICTURE SYSTEM 2 Diagnostic, or RS for PS2 Acceptance Test.

T indicates the Type of program it:is

T=C for Commands (i.e,, BATCH type operation files)
T=D for Diagnostic
T=P for Program

xxx indicates which diagnostic or program it is (xx=001-999)

is the filename/extension separator

Y is the type of file:

Y=A for ASCII (PS1 only)
Y=L for Load Absolute format (for Absolute Loader)
Y=M for MAPSIM statements (to be interpreted by MAPSIM---PS1 only)
Y=P for PICSYS statements (to be interpreted by PICSYS---PS1 only)

Y=s for Save image format (loadable by the Diagnostic Monitor)

zz is the version number of the file

NOTE: If Yzz=ESD this indicates an E&S Diagnostic System file. Under an RSX distribution, Yzz=TSK indicates the latest version of the task, corresponding to highest Sxx under ESD.

A.4 PS-2 STANDARD DIAGNOSTICS

FILENAME	EXPLANATION
QSD000	PS2 DIO Interface Diagnostics
QSD001 (1)	PS2 DMA Interface Diagnostics
QSD002	PICTURE SYSTEM Memory Diagnostics
QSD003	MAP: Status, Maintenance and Match Tests
QSD004	MAP: PROM/RAM & DOIT Register Test
QSD005	MAP: FIFO and Extend Register Test
QSD006	MAP: RSR Dispatch Test
QSD007	MAP: RSR Register Test
QSD008	MAP: Subroutine Call & Return Test
QSD009	MAP: Counter and Transpose Tests
QSD010	MAP: Data Storage Card Tests
QSD011	MAP: ALU Tests
QSD012	MAP: Normalize Sense Tests
QSD013	MAP: MRA and Normalizer Tests
QSD014	MAP: MRA Reciprocal ROM Tests
QSD015	MAP: Multiplier Tests
QSD016	MAP: Dispatch Code Tests
QSD017 (1)	MAP: Control ROM Test
QSD018 (1)	MAP: Output Formatter
QSD020 (2)	LG Visual Test
QSD026 (2)	Character Generator Visual Test
QSD027 (2)	Character Generator Visual Test
QSD028 (2)	Character RAM Visual Test
QSD031	Refresh Controller Diagnostic
QSD033 (1)	Character Generator RAM Test
QSD100 (1)	PS2 Interrupt Diagnostics
QSD102	PS2 Tablet Diagnostic
QSD103	PS2 Function Switch & Lights Diagnostic
QSD104	PS2 Keyboard Diagnostic
QSD105	PS2 Control Dial/Analog Card Diagnostic
QSD106 (1)	RRD Keyboard Diagnostic

FILENAME	EXPLANATION
QSD107	PS2 32-Bit Function Switches and Lights Diagnostic
QSD108	PS2 Light Pen Diagnostic
QSD109	Writeable Control Store Diagnostic
QSD110	PS2 RTI Diagnostic
QSD111	PS2 Multi-User Refresh Controller Register Diagnostic
QSD112	PS2 Multi-User Refresh Controller Micro Program Counter Diagnostic
QSD113	PS2 Multi-User Refresh Controller Micro Controller Diagnostic
QSD114	PS2 Multi-User Refresh Controller Refresh Function Diagnostic
QSD115	PS2 Multi-User Refresh Controller Refresh Error Diagnostic
QSD116	PS2 Multi-User Refresh Controller Device Control Diagnostic
RSD000 (1)	MAP Speed Tests
RSD001 (1)	MAP Function Tests
RSD002 (1)	DMA Speed Tests
RSD003 (1)	Refresh Modes Tests
RSD004 (1)	Character Generator/Line Generator Speed Tests
RSD005 (1)	MAP Drawing Modes Test
RSD006	Refresh Rates Test
RSD007 (2)	Line Generator Functions Test
RSD008 (2)	Scope Selection Test
RSD009	Character Generator Visual Test
RSD010 (2)	MAP Transformation Test (Version 1)
RSD011 (2)	MAP Transformation Test (Version 2)
RSD012 (2)	Peripherals Visual Test
ARCH2 (1)	System Integration Test
QSDDT	PS2 Debugging Program

---

(1) Not available under Mapped RSX-11M

(2) Available for restricted use under Mapped RSX-11M; Autorefresh phases only.

A.5 ESD SYSTEM AND UTILITY PROGRAMS

FILENAME.EXTENSION	EXPLANATION
ESDMON.ESD	E&S Diagnostic Monitor (RK05)
ESDTMN.ESD	E&S Diagnostic Monitor (DECtape)
RK .ESD	Disk Device Handler (RK05)
DT .ESD	DECtape Device Handler (TC11)
PR .ESD	Paper Tape Reader Device Handler (PC11 or PR11)
PP .ESD	Paper Punch Device Handler (PC11)
TT .ESD	Terminal Device Handler (ASR-33,LA-36,etc.)
UPDATE.S01	Diagnostic File Utility Program
PATCH .S01	Diagnostic File PATCH Program

A.6 PS1 DIAGNOSTICS

FILENAME.EXTENSION	EXPLANATION
PSREGS.SYM	P.S. Unibus Register Addresses
PSP002.S02	PICSYS+ Interactive Diagnostic Program
PSP003.S02	MAPSIM Interactive Diagnostic Program
PSC001.P02	PICSYS MAP Diagnostics Command Stream
PSC002.P02	PICSYS Display Diagnostics Command Stream
PSD001.P02	Load/Store
PSD002.P02	Single Row Matcon, Positive
PSD003.P02	Single Row Matcon, Negative
PSD004.P02	Single Element Matcon
PSD005.P02	Matcon With Overflow
PSD006.P02	Relative Draws
PSD007.P02	Moveto With Overflow
PSD008.P02	Viewport Mapping A
PSD009.P02	Viewport Mapping B
PSD010.P02	Viewport Mapping C
PSD011.P02	Viewport Mapping D
PSD012.P02	Viewport Mapping E
PSD013.P02	Viewport Mapping F
PSD014.P02	Viewport Mapping G
PSD015.P02	Various W Values
PSD016.P02	Lines
PSD017.P02	Clipping, X=W Plane
PSD018.P02	Clipping, S=-W Plane
PSD019.P02	Clipping, Y=W Plane
PSD020.P02	Clipping, Y=-W Plane
PSD021.P02	Clipping, Z=W Plane
PSD022.P02	Clipping, Z=0 Plane
PSD023.P02	Character & Status
PSD024.P02	Display Diagonal Line

(PS1 STANDARD DIAGNOSTICS CONTINUED)

FILENAME.EXTENSION	EXPLANATION
PSD025.P02	Display Square
PSD026.P02	Display Clipped Diamond
PSD027.P02	Display Dots
PSD028.P02	Display Characters
PSD029.P02	Display Features
PSD030.P02	Scope Select 1
PSD031.P02	Scope Select 2
PSD032.P02	Scope Select 3
PSD033.P02	Scope Select 4
PSD034.P02	Random Load/Store
PSD035.P02	Random Push/Pop
PSD036.P02	Random Matrix Concatenation
PSD037.P02	Refresh Buffer Address Test
PSD038.P02	Refresh Buffer Capacity Display
PSD039.P02	Hit Bit Test
PSD040.P02	Cursor Display
PSD041.P02	Refresh Buffer Random Data and Command Test
PSD042.P02	Refresh Buffer Checkerboard Test
PSD043.P02	Refresh Buffer Write Pointer Test
PSD044.P02	Refresh Buffer Write Pointer Addressing Test
PSD045.P02	Picture Processor RSR NO OP Test
PSD051.M01	MAP Initialization
PSD052.M01	Clip Store Registers Diagnostic
PSD053.M01	Transforamtion Matrix Register Diagnostic
PSD054.M01	Matrix Stack Register Diagnostic
PSD055.M01	Normalizer and ROM Table Diagnostic
PSD056.M01	Multiplication and Sign Correction Diagnostic
PSD057.M01	Bus and Selector Path Diagnostic
PSD058.M01	Temporary Matrix Registers Diagnostic
PSD100.S04	Picture Generator Diagnostic
PSD101.S02	Drawing/Character Diagnostic
PSD102.S02	Picture Generator and Color Picture Display Diagnostic
PSD103.S02	Real-Time Clock Diagnostic
PSD800.S02	Tablet Diagnostic
PSD810.S02	Control Dials Diagnostic (AD01)
PSD811.S02	Control Dials Diagnostic (AR11)
PSD820.S02	Function Swithes & Lights Diagnostic
PSD830.S02	Alphanumeric Keyboard Diagnostic
PSD840.S02	Lorgnette Diagnostic
ARCH .S01	Integrated System Test (The Architecture Program)
BLADE .S01	Integrated System Test (The Turbine Blade Program)

APPENDIX B
SUMMARY OF E&S DIAGNOSTIC MONITOR COMMANDS

The following commands are available to the user of the E&S Diagnostic Monitor.

*DATE dd-mmm-yy(CR) or *DA dd-mmm-yy(CR)	Enters the current date (day-month-year).
*ASSIGN dev adev(CR) or *AS dev adev(CR)	Assigns dev a 1-3 character alternate device name (adev).
*RUN dev:filename.ext(CR) or *RU dev:filename.ext(CR)	Loads and begins execution of the indicate file from the specified device.

APPENDIX C

SUMMARY OF THE E&S DIAGNOSTIC FILE UTILITY PROGRAM OPERATIONS

The following switches are available to the user of the E&S Diagnostic Utility, UPDATE:

/A Copies file(s) in ASCII mode.

/B Copies file(s) in formatted Binary mode.

/C Compresses device directory and files.

/D Deletes file(s) from specified devices.

/I Copies file(s) Individually rather than as a collective group (PIP /X).

/L Lists the directory of the specified device.

/O Boots the specified device (RK0: or DT0: only).

/W Allows Diagnostic Monitor system files (those with ESD extention) to be operated on (With permission) (PIP .SYS, /Y).

/Z Zeros (initializes) the directory of the specified device allocating 8 disk blocks for the directory size.

/Q Typed on a line by itself, enables the switches which follow.

/E Lists the entire directory including unused spaces and their sizes.

/F Prints the short directory (file names only) of the specified device.

/G Ignores any input errors which occur during a file transfer and continues copying.

/N Used with /Z to specify the number of directory blocks to allocate to the directory.

/R Renames the specified file.

/T Extends the number of blocks allocated for a file.

/U Copies the specified bootstrap file into absolute blocks Ø and 2 of the specified device.

The following commands are available to the user of the E&S Diagnostic Utility, PATCH:

n/ Print contents of location n.

<carriage return>
Close current location.

<line feed>
Close the current location and open the next word.

↑ Close the current location and open the previous word.

@ Close the current location and open the word addressed by it.

E Close the currently open file and exit to the diagnostic monitor.

F Close the currently open file and request a new filename to be input.

APPENDIX D
INTRODUCTION TO RSX OPERATION

A complete set of PS-2 Diagnostics is distributed under uic [220,15] to run under Unmapped RSX-11M. Also, a subset of the PS-2 Diagnostics is distributed under uic [220,14] to run under Mapped RSX-11M to avoid, when possible, shutting down a multi-user system while diagnosing the Picture System. The diagnostics and phases of diagnostics which may run under Mapped RSX are those which do not use DMA or interrupts. For a complete list, see Appendix A. For information concerning the transfer of RSX PS-2 Diagnostics to the system device, see section 1.1 and 1.4.

RSX Distribution of the PS-2 Diagnostics require a minimum of 28K words of memory.

D.1 STARTUP SEQUENCE

When the RSX-11M system device has been bootstrapped, the following sequence should occur, with the operator supplying the time and date and "UIC" command as indicated by underlining:

```
RSX-11M B03 BL18 XXK
>RED XX0:=SY0:
>MOU XX0:UNMOBJ
>@[1,2]STARTUP
>*PLEASE ENTER TIME AND DATE (HH:MM MM/DD/YY) [S]: XX:XX XX/XX/77
>TIM
>XX:XX: XX/XX/77
>@<EOF>
>SET /UIC-[XXX,XXX]
>
```

This system is now prepared to run PS-2 diagnostics as requested by the operator.

D.2 INITIATION OF TASKS

Tasks are initiated by means of the "RUN" command, as in the following examples:

>RUN QSD001 or
>RUN QSD001.TSK

These two examples are equivalent; both examples request the latest version of the task (program) QSD001. To run a task version other than the latest, use a command of the following form:

>RUN QSD001.S01

D.3 TERMINATION OF TASKS

Some difficulty may be encountered in this area by users who are unfamiliar with RSX, yet familiar with the E&S Diagnostic Monitor. The customary "control-C" will not terminate a diagnostic program under RSX, except in a few special cases.

Two major cases must be distinguished: (a) the more prevalent case, in which a diagnostic program is running, but not awaiting input from the operator's terminal, and (b) the case in which a task is awaiting input from the operator's terminal. In the latter case, all diagnostics will interpret "control-Z" as a command to terminate execution.

When any task is running but not requesting input, any input from the operator terminal "wakes up" the RSX Monitor (MCR), but does so without suspending the active task (presumably, in this case, a diagnostic). There are two reasons for this: (a) the operating system is multiprogramming, allowing the active task to continue while MCR awaits a complete line of input, and (b) the operator may

desire to initiate another task or query the system without "aborting" the already active task.

The action taken by MCR, in the event of operator input, is as follows: interpret the next line of input as an MCR command string; execute it (or output an error message); allow any tasks which may now be active to continue execution.

The method for termination of an active diagnostic which is not seeking operator input is as follows: (a) Type "A", and wait for the "A" to be echoed (printed) on the terminal. (b) Type "BO" and carriage return. This passes the command "ABO" (meaning: abort the currently active task) to MCR. It is possible, if one types this line too quickly, to lose one or more characters at the beginning of the command string.

Two pitfalls to be avoided are the following: (a) Typing "control-C" when the diagnostic is not awaiting input. This "wakes up" MCR, but does not terminate the diagnostic. MCR is now awaiting completion of an input line (signalled by carriage-return) --- but the input line will be syntactically invalid as it begins with "control-C". (b) If a "RUN" command is issued when the preceding task is still active, no error message will occur. MCR would actually allow both diagnostic tasks to run concurrently, were it not for the fact that they must run in the same memory partition. In this case, the second task would be put on a list of tasks awaiting activation, and the preceding task would still be active.

D.4 MODIFYING PS-2 DEVICE AND INTERRUPT ASSIGNMENTS

This section discusses the procedure for "patching" or permanently modifying the PS-2 Diagnostics Device and interrupt

assignments under RSX. This discussion should be considered a supplement to the treatment of the same subject (under ESD) in Chapter 5.

Patches are accomplished under RSX by running the "ZAP" utility. The first command after running ZAP names a task file to be modified, for example:

```
>RUN $ZAP(CR)
ZAP>QSD001(CR)
                (cue character produced by ZAP)
_
```

Locations in the task image are referenced by a segment number (always "2" in the case of the diagnostics) followed by a colon and the absolute address. The basic ZAP commands required are as follows, where "nnnnnn" designates an absolute address, and "dddddd" designates a 16-bit data value:

```
2:nnnnnn/       (open and display absolute location nnnnnn)
dddddd (CR)     (deposit dddddd in the open location)
X(CR)           (Exit; terminate)
```

Beginning locations of the Host Computer Table (H) and the PICTURE SYSTEM Table (P) are as follow:

	Unmapped	Mapped
P Table first entry (P1):	2:41310	2:1330
H Table first entry (H1):	2:41552	2:1572

Note: The addresses of these locations are subject to slight change.

D.5 ARCH2 OPERATION

The following are non-standard features of ARCH2 (UIC=[200,200]) operation under Unmapped RSX-11M:

(a) A maximum of 12 objects may be placed on the platform, as opposed to the standard maximum of 30.

(b) Error messages are not output by the unmapped RSX version of ARCH2. Such messages (e.g. "ERROR 2 DETECTED IN GRAPHICS SUBROUTINE Ø") would typically indicate that the PS-2 Interface is not functioning properly.

Termination of ARCH2: The system will crash when ARCH2 is aborted, unless the I/O Status Word (IOST) is cleared to disable Picture System interrupts prior to abortion of the task, as in the following sequence:

```
>OPEN 76767Ø(CR)
xxxxxx Ø (Altmode)
>ABO(CR)
```

D.6 OPERATION OF MAPPED PS-2 DIAGNOSTIC TASKS

The Mapped PS-2 Diagnostic Tasks communicate with the PS-2 interface registers directly, rather than by means of a driver. This creates a possibility of conflicts between diagnostics and user software concurrently communicating with the Picture System. To ensure that no other user is now attempting to communicate with the Picture System, and to prevent access by user software once the diagnostics have begun to execute, one should allocate the Picture System, or all constituent work stations in the case of a Multi-user Refresh Controller, by means of the "ALL" command. When no further use of the diagnostics is required, the Picture System or work stations should be deallocated by means of the "DEA" command.

D.7 RSX REFERENCES

The following manuals by DEC are recommended as the first recourse for persons who wish to become more familiar with RSX, or who encounter problems which have not been dealt with in this manual:

RSX-11M/RSX-11S DOCUMENTATION DIRECTORY:
A guide to the available Documentation on RSX.

INTRODUCTION TO RSX-11M:
A conceptual introduction.

RSX-11M OPERATOR's PROCEDURES MANUAL:
Operational information including MCR (monitor) commands.

RSX-11 UTILITIES PROCEDURES MANUAL:
Concerning the use of ZAP, PIP, and other utilities.

PDP-11 PRESERVATION UTILITY (PRESRV) USER'S GUIDE:
For problems encountered in transferring the system image from Magtape to the system device.

RSX-11M SYSTEM GENERATION MANUAL:
The section entitled "Bootstrapping the Baseline System" includes bootstrap procedures.

APPENDIX E
PSDEVØ GENERATION AND MODIFICATION

The following information amplifies Section 1.5, "Transferring Diagnostic Tasks from Magtape to Disk (Mapped RSX-11M)".

PSDEVØ.MAC contains the following:

```
.TITLE PSDEVØ
.IDENT /NC/
.PSECT PSDEVØ,D,GBL,OVR
.BLKB 72
.END
```

PSDEVØ.CMD contains the following TKB commands:

```
PSDEVØ/PI/-HD,,PSDEVØ=PSDEVØ
/
STACK=Ø
PAR=PSDEVØ
//
```

The address of the PS-2 Interface Register Block (normally 767660) is derived from two sources of information. The set /MAIN command determines the 100_8 byte block which corresponds to location Ø in PSDEVØ (normally SET /MAIN=PSDEVØ: 7676:1:DEV for 767600). The contents of HSTB location 1

(H1) minus 160000 determines the offset from the base of PSDEVØ to the beginning of the Interface Register Block (Normally H1 = 160060 for offset = 60). These parameters, and the size of PSDEVØ, if necessary, may be modified to establish a non-standard Picture System Unibus base address.

APPENDIX F

PS-2/INTERDATA 8-32 INTERFACE REGISTERS

Following are the legal values for the register selection parameter of the "READ" and "WRITE" subroutines in the Interdata PSIO module. These values also represent the addresses of the registers in "I" addressing mode in QSDDT. For more information about these registers, consult the PS-2 Interdata 8/32 Reference Manual.

ADDRESS	MNEMONIC	MEANING	INTERDATA INSTRUCTION
Ø	PSDATA	Picture System Data	RH or WH X'AØ'
1	DIOPSA	DIO Address	RH or WH X'A1'
4	PS	PS Control/Status	SS or OC X'AØ'
5	PSDIO	Direct I/O Control/ Status	SS or OC X'A1'
6	PSDMA	DMA Control/Status	SS or OC X'A2'
10	DMABA	DMA Beginning Address	RH or WH X'A2'
11	EXTBA	Extended DMA Beginning Address	RH or WH X'A2'
12	DMAEA	DMA End Address	RH or WH X'A2'
13	EXTEA	Extended DMA End Address	RH or WH X'A2'
14	MAINT	Maintenance Control	RH or WH X'A3'

APPENDIX G

PS-2/NORD-10 INTERFACE REGISTERS

Following are the legal values for the register selection parameter of the "READ" and "WRITE" subroutines in the NORD-10 PSIO module. These values also represent the "I" addressing mode in QSDDT. For more information, consult the NORD-10 Interface documentation.

ADDRESS	MNEMONIC	MEANING
Ø	DIOPSA	DIO PS Address
1	PSDATA	DIO Data
2	PSDIO	DIO Control/Status
3	DMABA	DMA Computer Memory Address
4	PSDMA	DMA Control/Status
5	DMAWC	DMA Word Count

APPENDIX H
HOST TABLE (HSTB) AND PICTURE SYSTEM TABLE (PSTB)

TABLE H-1
HOST COMPUTER Device and Interrupt Addresses

Entry	Contents	Description
1	167660	PDP-11 Base
2	340	PDP-11 Interrupt Vector Base
3-5		Reserved for RTI Diagnostics

TABLE H-2

PICTURE SYSTEM Device and Interrupt Addresses

Entry	Contents	Description
1		Reserved
2	177744	Real Time Clock Base
3	177740	Picture Generator #1 (PG #1) Base
4	177775	PG #1 Passive Input Port
5	177700	PG #2 Base
6	100000	PG #2 Passive Input Port
7	177730	Refresh Controller #1 (RC #1) Base
10	177762	System-Interrupt Request Register for RC #1 (SYSREQ-RC 1)
11	4	Shift into SYSREQ-RC 1 for RC #1 Interrupt
12	177670	RC #2 Base
13	100000	SYSREQ for RC #2 (SYSREQ-RC2)
14	0	Shift into SYSREQ-RC2 for RC #2 Interrupt
15	177750	Picture Processor #1 (PP #1) Base
16	177777	PP #1 Passive Input Port
17	177776	PP #1 Passive Output Port
20	177762	SYSREQ for PP #1 (SYSREQ-PP1)
21	0	Shift into SYSREQ-PP 1 for PP #1 Interrupt
22	177710	PP #2 Base
23	100000	PP #2 Passive Input Port
24	100000	PP #2 Passive Output Port
25	100000	SYSREQ for PP #2 (SYSREQ-PP2)
26	0	Shift into SYSREQ-PP2 for PP #2 Interrupt
27	177747	DMA #1 Base
30	177770	DMA #1 Passive Input Port
31	177746	DMA #2 Base
32	100000	DMA #2 Passive Input Port
33	177664	Tablet Base
34	177764	Device-Interrupt Request Register for Tablet (DEVREQ-TAB)
35	0	Shift into DEVREQ-TAB for Tablet Interrupt
36	177607	Keyboard Base
37	177764	DEVREQ for Keyboard (DEVREQ-KBD)

Entry	Contents	Description
40	4	Shift into DEVREQ-KBD for Keyboard Interrupt
41	177626	Function Switch-and-Lights Base
42	177764	DEVREQ for Function Switches-and Lights (DEVREQ-FSL)
43	10	Shift into DEVREQ-FSL for Function Switches Interrupt
44	177720	Light Pen Base
45	177762	DEVREQ for Light Pen (DEVREQ-LPEN)
46	7	Shift into DEVREQ-LPEN for Light Pen Interrupt
47	177500	A-to-D Converter Base
50	177772	RTI passport
51	100000	Unused
52	177730	RF Base
53	177762	RF #1 Interrupt Base
54	4	RF #1 Interrupt Shift
55	177440	32-Bit FSWL #1 Base
56	177764	32-Bit FSWL #1 Interrupt Base
57	10	32-Bit FSWL #1 Shift Count
60	177660	Tablet #2 Base
61	177764	Tablet #2 Interrupt Base
62	1	Tablet #2 Interrupt Shift Count
63	177606	Keyboard #2 Base
64	177764	KB #2 Interrupt Base
65	5	KB #2 Interrupt Shift Count
66	177624	Function Switches-and-Lights #2 Base
67	177764	FSWL #2 Interrupt Base
70	11	FSWL #2 Interrupt Shift Count
71	100000	Light Pen #2 Base
72	100000	LP #2 Interrupt Base
73	100000	LP #2 Interrupt Shift Count
74	177520	A-to-D Converter #2 Base
75	100000	RTI #2 Passive Input Port
76	0	General Purpose Software Parameter
77	0	General Purpose Software Parameter

Entry	Contents	Description
100	0	General Purpose Software Parameter
101	0	General Purpose Software Parameter
102	177450	32 FSWL #2 Base
103	177764	32 FSWL #2 Interrupt Base
104	11	32 FSWL #2 Interrupt Shift Count
105	177670	Multi-User Refresh Controller (RF #2) Base
106	100000	RF #2 Interrupt Base
107	100000	RF #2 Interrupt Shift Count
110	100000	Unused
111	100000	Unused
112	100000	Unused
113	100000	Unused
117	100000	Unused

APPENDIX I
PS-2 SYSTEM CONTROL BLOCK

The System Control Block contains all the PS-2 device control and maintenance registers. Following is a list of the addresses of all such registers.

FROM	TO	DEVICE	REGISTER NAME	REMARKS
177400	177437	Unused		
177440		FSL1	FSWR0	32 Button Box
177441		FSL1	FSWR1	
177442		FSL1	FSWR2	
177443		FSL1	FSWR3	
177444		FSL1	FSLR0	
177445		FSL1	FSLR1	
177446		FSL1	FSLR2	
177447		FSL1	FSLR3	
177450	177477	FSL2-4		
177500		A/D1	ADDR0	Control Dials/ Joystick
177501		A/D1	ADDR1	
177502		A/D1	ADDR2	
177503		A/D1	ADDR3	
177504		A/D1	ADDR4	
177505		A/D1	ADDR5	
177506		A/D1	ADDR6	
177507		A/D1	ADDR7	
177510		A/D1	ADDR10	
177511		A/D1	ADDR11	
177512		A/D1	ADDR12	
177513		A/D1	ADDR13	
177514		A/D1	ADDR14	
177515		A/D1	ADDR15	
177516		A/D1	ADDR16	
177517		A/D1	ADDR17	

FROM	TO	DEVICE	REGISTER NAME	REMARKS
177520	177577	A/D2-4		
177600	177606	KB8-2	KBDATA	Keyboards
177607		KB1		
177610	177625	FSL8-2		Function Switches & Lights
177626		FSL1	FSWR	
177627		FSL1	FSLR	
177630	177663	DT8-2		Data Tablets
177664		DT1	DTSR	
177665		DT1	DTXDAT	
177666		DT1	DTYDAT	
177667		DT1	DTZDAT	
177670	177677	RF2	Same as RF1	Refresh Controller
177700	177703	PG2	Same as PG1	Picture Generator
177704	177707	Unused		
177710	177717	PP2	Same as PP1	Picture Processor
177720		LP1-4	LPSR	Light Pens
177721		LP1-4	LPSN	
177722		LP1-4	LPSCT	
177723		LP1-4	LPTCT	
177724		LP1-4	LPSX	
177725		LP1-4	LPSY	
177726		LP1-4	LPEX	
177727		LP1-4	LPEY	
177730		RF1	RFCSN	Refresh Controller
177731		RF1	RFSN	
177732		RF1	RFAWA	
177733		RF1	RFAWL	
177734		RF1	RFAIA	
177735		RF1	RFASA	
177736		RF1	RFAIL	
177737		RF1	RFSR	
177740		PG1	PGSR	Picture Generator
177741		PG1	PGXBUS	
177742		PG1	PGYBUS	
177743		Unused		

FROM	TO	DEVICE	REGISTER NAME	REMARKS
177744		RTC	RTCCNT	Real-Time Clock
177745		RTC	RTCSR	
177746		DMA2	DMAPSA	DMA2 PS Address
177747		DMA1	DMAPSA	DMA1 PS Address
177750		PP1	MAOL	Picture Processor
177751		PP1	MAOA	
177752		PP1	MAIA	
177753		PP1	MSR	
177754		PP1	MMSR	
177755		PP1	MMRSR	
177756		PP1	MMPAR	
177757		PP1	MMBUS	
177760		RTC	RTCREQ	Real-Time Clock Interrupt
177761		RTC	RTCIE	
177762		SYSTEM	SYSREQ	System Interrupts
177763		SYSTEM	SYSIE	
177764		DEVICES	DEVREQ 0-15	Device Interrupts
177765		DEVICES	DEVIE 0-15	
177766		DEVICES	DEVREQ 16-31	
177767		DEVICES	DEVIE 16-31	
177770		DMA1	DMAPIP	DMA Passive Input Port
177771		Reserved for options using		Passive Ports
177772		RTI	RTIP	RTI Passive Input Port
177773	177774	Reserved for options using		Passive Ports
177775		PG1	LGPIP	LG Passive Input Port
177776		PP1	MPOP	MAP Passive Output Port
177777		PP1	MPIP	MAP Passive Input Port