

IDENTIFICATION

Product Code: MAINDEC-11-DCQKC-D-0
Product Name: 11/42 and 11/45 INSTRUCTION EXERCISER
Date Created: SEPTEMBER 21, 1974
Maintainer: Diagnostic Group
Author: John Adams

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation. Digital Equipment Corporation assumes no responsibility for any errors that may appear in this manual.

The software described in this document is furnished to the purchaser under a license for use on a single computer system and can be copied (with inclusion of DIGITAL's copyright notice) only for use in such system, except as may otherwise be provided in writing by DIGITAL.

Digital Equipment Corporation assumes no responsibility for the use or reliability of its software on equipment that is not supplied by DIGITAL.

COPYRIGHT (c) 1973, 1974

DIGITAL EQUIPMENT CORPORATION

ABSTRACT

Chapter 1 REQUIREMENTS

- 1.1 EQUIPMENT
 - 1.1.1 Optional Equipment Used
- 1.2 STORAGE
- 1.3 PRELIMINARY PROGRAMS

Chapter 2 LOADING AND STARTING PROCEDURE

- 2.1 ACT11 OPERATION

Chapter 3 SWITCH SETTINGS

- 3.1 11/45 DISPLAY REGISTER

Chapter 4 ERRORS

- 4.1.1 Error Printout Format (CP Error)
- 4.1.2 Error Printout Format (Device Error)
- 4.1.3 Error Printout Format (Parity Error)
- 4.1.4 Error Printout Format (Relocation Error)
- 4.2 PARITY ERROR DETECTION
- 4.3 ERROR LOOPING
- 4.4 UNPREDICTED ERRORS
- 4.5 TRAP TO LOCATION 4
- 4.6 TRAP TO LOCATION 10
- 4.7 MEMORY MANAGEMENT (KT11) ABORT
- 4.8 ERROR DISCUSSION

Chapter 5 SUBROUTINE ABSTRACTS

- 5.1 SCOPEA
- 5.2 ERROR
- 5.3 PROGRAM RELOCATION
 - 5.3.1 RELOC
 - 5.3.2 Relocation above 28K (STMM)
 - 5.3.3 IODEV
 - 5.3.4 WAITIO
- 5.4 CLOCK INTERRUPT

5.5 END

Chapter 6 MISCELLANEOUS

- 6.1 EXECUTION TIME
- 6.2 PASS MODIFICATION
- 6.3 I/O DEVICE ADDRESS MODIFICATION
- 6.4 memory modification
- 6.5 user defined relocation limits

Chapter 7 PROGRAM DESCRIPTION

- 7.1 STACK POINTER
- 7.2 POWER FAILURE

ABSTRACT

This Diagnostic Program is designed to be a comprehensive check of the PDP-11/40 and PDP-11/45 processors. The program executes each instruction in all address modes and includes tests for traps and the Teletype interrupt sequence. The program relocates the test code throughout memory 0-124K. If selected, the program may be relocated by any of the available disks.

CHAPTER 1
REQUIREMENTS

1.1 EQUIPMENT

PDP-11 Family Central Processor with 8K memory.

1.1.2 Optional Equipment Used

1. KW11=P (Programmable Clock)
2. KW11=L (Line Frequency Clock)
3. All parity memory options
4. KT11=C,D (11/40, 11/45 Memory Management)
5. RK11, RF11, RP11, RS03/4, RC11
6. KJ-11 (11/40 Stack Limit)
7. EIS (11/40 Extended Instruction Set)

1.2 STORAGE

The program loads into the first 6K of memory, and runs in all memory (exclusive of loaders).

1.3 PRELIMINARY PROGRAMS

None.

CHAPTER 2 LOADING AND STARTING PROCEDURE

Load the program using the Absolute Loader. If console TTY is a serial device (LA30S, VT05, etc.), filler characters are required. Deposit into location 1002 (FILLS) a 0 (the filler character) and location 1003 11(Octal) (the filler count).

Load Address = 200
Press start,
Set operating switches

Contents of OPT,CP is typed on first pass (see Chart 7)
(Initial load)
Pass count is printed after each pass (see Section 5.5)
"DCQKC DONE" is printed when done (see Section 6.1).

If no console TTY is available, set SW15=1 (HALT ON ERROR).

2.1 ACT11 OPERATION

If the program is run in quick verify mode, no subtest iterations are performed.

CHAPTER 3
SWITCH SETTINGS

SW15	HALT ON ERROR	This switch when set will halt the processor when an error is detected. The PC+2 and the current status at the time of the error is stored on the stack (R6). If this switch is set before an error is detected, the program halts as described above. The program may be halted after the error timeout occurs by setting SW15 after the timeout begins.
SW14	LOOP SUBTEST	This switch when set loops the current subtest running regardless of error.
SW13	INHIBIT ERROR PRINTOUT	This switch when set inhibits the error printout.
SW12	INHIBIT RELOCATION	This switch when set causes the program to be executed only in the first 8K of memory. This switch cannot be set when the program is running.
SW11	INHIBIT SUB-TEST ITERATION	This switch when set inhibits subtest reiteration. Normally each subtest is executed 8 times before the next subtest is run. Setting SW11 causes each test to be executed once before starting the next subtest.
SW10	RING BELL ON ERROR	This switch when set will ring the bell when an error is detected.
SW9	INHIBIT RELOCATION	This switch when set inhibits relocation of the program above 28K.
SW8 SW7=0	LOAD PDP-11/45 MICRO BREAK	This switch when set loads the micro break register with the value set into

REGISTER SW7=0 at the beginning of each subtest.

SW7 INHIBIT END OF PASS TYPEOUT This switch when reset inhibits the end of pass typeout (The Quick Brown Fox,...).

SW6 INHIBIT CLOCK INTERRUPTS This switch when set will turn the clock(s) off.

SW05 ENABLE RELOCATION VIA ALL AVAIL. DISKS This switch will cause program relocation via all available disks Round Robin style, i.e., first relocation via CP, then RK, RF, RP, etc.

SW04 ENABLE RANDOM DISK ADDRESS SELECTION FOR RELOCATION If not enabled all disk relocation transfers begin at disk address 0.

SW03 ENABLE RELOCATION VIA I/O DEVICE

SW02-SW00 DEVICE CODES These switches when set cause the program to relocate the test code using the device specified below:

Value	Device
0	CP
1	RK
2	RF
3	RP
4	RC
5	DO NOT USE
6	RS04
7	CP

NOTE

When relocating via an I/O device, set in the value to select the device then set switch 3.

3.1 11/45 DISPLAY REGISTER

The pass count is displayed in bits 00-02. The section number is displayed in bits 06-03. The most significant byte of the base address (contents of FRSTAD) of the section of code being executed is displayed in bits 15-08. When memory management is enabled the contents of KIPAR2 is displayed. KIPAR2 contains the base page

address of the code being executed.

NOTE

The RF11 Data Buffer Register also displays the above information if the RF is selected.

CHAPTER 4
ERRORS

If an error is detected, the program will wrap to the Error Handling Routine (ERROR). If error timeout is enabled, this routine will type the PC and the processor status at the time of the error. Also, (if required), the original PC (where the PC was relocated from).

4.1.1 Error Printout Format (CP Error)

PASS # AAAA VPC=BBBBBB PSW=DDDDDD

or

PASS # AAAA VPC=BBBBBB PSW=DDDDDD RPC=CCCCC

or

PASS # AAAA VPC=BBBBBB PSW=DDDDDD PPC=EEEEEE

where:
VPC=Virtual PC
RPC=PC of original code
PPC=Physical PC
AAAA=PASS COUNT
BBBBBB=Virtual PC at the time of the error
CCCCC=PC of the original code relocated
DDDDD=PSW at the time of the error
EEEEEE=Physical PC at the time of the error.

The first error format shows an error detected when the program is not relocated, and, in this instance VPC=PPC. The error is probably a CP error.

The second error format show an error detected when the program is relocated below 28K, and, in this instance VPC=PPC. The error is probably due to a memory error.

The third error format shows an error detected when the program is relocated above 28K. The error is probably due to a memory error. Note that VPC is the PC of the original code.

To obtain the 'Physical' PC (11/45 only), set the address selector to the KLI position. Load address and examine the PC address, set the address selector to 'Program Physical'. The address displayed is the Physical PC. On the 11/40 to obtain the 'Physical' PC add the contents of KIPAR2 or KIPAR3 to the Virtual PC.

NOTE

Use caution when examining/depositing into addresses when memory management is enabled.

4.1.2 Error Printout Format (Device Error)

PASS # AAAA VPC=BBBBBB XX ERROR

111111 222222 333333 444444 555555 666666

where: VPC=Virtual PC
AAAA=Pass count
BBBBBB=Virtual PC at time of error
XX=two letter device identifier
111111-666666=Contents of device register

4.1.3 Error Printout Format (Parity Error)

Parity Error

The PC at the time of the error is typed as shown in Section 4.1.1.

Memory Address = XXXXXX, Good Data = XXXXXX, Bad Data = XXXXXX.

NOTE

The address typed is the 18 bit physical address.

4.1.4 Error Printout Format (Relocation Error)

PASS # AAAA VPC=BBBBBB MM ERROR

FROM ADRS=XXXXXX DATA=XXXXXX TO ADRS=XXXXXX DATA=XXXXXX

NOTE

The addresses are 18 bit physical addresses "from" address is in R0 "to" address is in R2.

4.2 PARITY ERROR DETECTION

If a parity error is detected the program will type a message "PARITY ERROR". Print the PC at the time of the error (via HLT) and scan memory for the parity error. When the program finds the parity error it will type a message "MEMORY ADDRESS IS BBBBBB". When the address is found the failing address is scanned with a binary count pattern. When the program finds the failing data the good data and bad data are typed. If the program does not find the parity error on the address/data scan it will type a message "PARITY ERROR NOT DETECTED ON ADDRESS/DATA SCAN". The program is then restarted.

4.3 ERROR LOOPING

The subtest detecting the error may be looped indefinitely by setting SW14. Setting SW13 will inhibit the timeout and allow scooping the faulty signal(s).

4.4 UNPREDICTED ERRORS

The program may on occasion detect a memory error the results of which were not predictable in which case the program may behave unpredictably. When this happens the user must retrace the program steps to resolve where the error occurred. The following items should be considered and may be of use when retracing a failure of this nature.

1. Halt the program (if necessary).
2. Examine RELR1 (1006) contains the unrelocated value of the PC of the last test that was successfully executed.
3. Examine FACTOR
Address FACTOR (1004) contains the relocation factor.
4. Examine all locations starting with the address specified in R1/R11 (if PSW BIT11 = 0/1) comparing their contents with the contents of the corresponding unrelocated code (specified in

1006) as shown in the listing. Examine and compare until either a difference in instruction (i.e., the error) or the next 'scope' is seen.

- 1A. Examine the stack (R6)
The top word on the stack contains the PC at the time of the trap. If the PC is greater than the last location in the listing then =
- 2A. Examine location 1004 (FACTOR)
This location contains the program relocation factor which, when subtracted from the PC gives the PC of the original code.

4.5 TRAP TO LOCATION 4

If a trap to location 4 occurs the program will type: "TRAP TO 4". Then the error printout information (as in 4.1.1) will be typed.

NOTE

The PC typed will be the PC-2 at the time that the trap occurred. The program will then restart at the beginning (START).

4.6 TRAP TO LOCATION 10

If a trap to location 10 (reserved instruction) occurs the program will type: "RESERVED INSTRUCTION TRAP" and the additional information (as in 4.1.1) the PC typed will be the PC-2 at the time of the trap. The program will restart at the beginning (START).

4.7 MEMORY MANAGEMENT (KT11) ABORT

If a KT11 abort (trap at 250) occurs, the program will type a message "KT11 ABORT". Then the error printout information (as in 4.1.1) will be typed.

NOTE

The PC typed will be the contents of SR2 at the time that the trap occurred. The program will then restart at the beginning (START).

4.8 ERROR DISCUSSION

An error detected when the program is not relocated is likely to be a CP malfunction. An error detected when the program is relocated between 40000 and 160000 could be either a CP or memory malfunction. An error detected when the program is relocated above 160000 (28K) is most likely a memory malfunction. The memory exerciser (DZQMB-) should be run if a memory failure is suspected, selecting only those bank(s) deemed bad.

CHAPTER 5
SUBROUTINE ABSTRACTS

5.1 SCOPEA

The SCOPEA Routine is entered by the scope (EMT) instruction and is executed at the start of each subtest. The routine monitors SW14, SW11 and SW8 and takes appropriate action. Also, this routine stores in R1/R11 the first address of the subtest being entered.

5.2 ERROR

The Error Routine is entered by the HLT (trap) instruction, and is executed when a predictable error is detected. This routine monitors SW15, SW13, and SW10.

5.3 PROGRAM RELOCATION

Four routines are used to perform program relocation. The general flow is as follows:

If below 28K The RELOC Routine is called after a section of code has been executed. If an I/O device is selected, subroutine IODEV is called and the routine WAITIO is executed while the device is transferring code.

If above 28K The STMM Routine is called after the entire program has been executed. If an I/O device is selected, subroutine IODEV is called and the routine WAITIO is executed while the device is transferring code.

5.3.1 RELOC

The RELOC Routine is entered by a MOV #RELOC,PC instruction. This routine relocates the program code throughout memory and 'jumps' to the relocated code after it has been moved successfully. The code is relocated by 'moving' the code via MOV instructions. If an I/O device is selected via switch register <3=0>, the code is relocated by writing the code onto the I/O device and reading the code back into its relocated position. If the code cannot be relocated (because of insufficient memory) the routine 'jumps' to the next section of unrellocated program code. The code moved is less than 1K (4000) bytes). At the start and end of each section of code to be moved is a section of code which establishes the first address of the code to be moved, and sets a scope pointer (R1/R11) and also a section which establishes the last address and 'jumps' to the relocation (RELOC) routine. Each section of code is identified as shown below,

```
1000000000FIRST ADDRESS TO BE RELOCATED0000000000
```

```
CODE TO BE MOVED AND EXECUTED
```

```
1000000000LAST ADDRESS OF CODE TO BE RELOCATED0000000000
```

THE RELOC Routine does not relocate program code into the last 1000(octal) bytes of memory, thus preserving the loaders. This routine monitors SW12, SW05, and SW03.

5.3.2 Relocation above 28K (STMM)

The STMM Subroutine relocates the program code above 28K if memory and the KT option are available. The routine moves the code at 0-8K upwards to addresses above 28K. Each succeeding relocation is to memory 1K greater than the last. The program is executed in all cases from virtual memory addresses 0-37776, however, the physical address changes by 1K (4000) on each relocation.

NOTE

The 'Virtual' light (11/40) will be on when the program is executing above 28K.

This routine monitors SW12, SW09, SW05, and SW03.

5.3.3 IODEV

The IODEV Subroutine is called from either the RELOC or STMM routines whenever an I/O device is selected to perform program relocation. This routine obtains the physical BUS address for read and write and

the byte count from the calling routine. The device to be used is obtained from location DEV. The code to be relocated is written from its present position and then read into the relocated position. If a device error occurs the error is reported and the operation is retried up to three times.

5.3.4 WAITIO

The purpose of the WAITIO Routine is to reference via the CP the same memory locations as the device during the NPR transfers.

5.3.5 DSKADR

The DSKADR Subroutine is called from the IODEV routine. It generates a random disk address for the selected disk if SW04 is set. Otherwise it generates a '0' disk address. The generated random addresses are limited (so disk overflow will not occur) by the table ADRTA3.

5.4 CLOCK INTERRUPT

The Clock Interrupt for the line and programmable clocks increment locations LTICKS and PTICKS on each interrupt. This routine monitors SW06.

5.5 END

This routine is entered at the completion of each pass. It sets up (loads new processor status) for the next pass and prints an end of pass message:

```
THE QUICK BROWN FOX JUMPED OVER THE LAZY DOGS BACK  
0123456789 PASS # AAAA
```

CHAPTER 6 MISCELLANEOUS

6.1 EXECUTION TIME

The execution time is highly variable (dependent on processor, type of memory, and amount of memory), however, when the program is running successfully there is a noticeable 'Flicker' displayed on the console light pattern. The 'Flicker' will dim when 'T' bit trap passes (every odd pass) are running, the program should be run for a minimum of:

4 PASSES (PASS # 0003) 11/40
8 PASSES (PASS # 0007) 11/45

some typical times follow:

PDP-11/45 WITH 104K MEMORY (96K CORE, 8K MOS)-24 MINS
PDP-11/45 WITH 48K MEMORY-10 MINS

6.2 PASS MODIFICATION

The PSW of the pass may be modified by patching into location PSWTAB+2 the desired PSW. For example patching 040000 into PSWTAB+2 causes the program to run in supervisor mode on the second pass.

6.3 I/O DEVICE ADDRESS MODIFICATION

To modify the program address of the I/O devices on the unibus patch the appropriate device table (see listing table of contents - Device Tables) and also the appropriate table entry at 'REGADR' in the error service routine.

6.4 MEMORY MODIFICATION

The program may be modified to provide extended memory exercising. Essentially the modification increases the test iteration count which causes test code to be executed in memory for a longer period of time. Note that this modification will increase the run time substantially. The modification is:

PATCH	LOCATION	FROM	TO
	5454	020040	100200

6.5 USER DEFINED RELOCATION LIMITS

The program will request a lower and upper limit for relocation. The limits must be between the last location in the listing and 157776. The program will execute in the lower 4K (0-17776) and the limits specified. The starting address is 204. To retain previously specified limits, start at 210.

CHAPTER 7

PROGRAM DESCRIPTION

The program is divided into four sections of position independent relocatable test code. Each section is approximately 1K words long, (except section B and A).

Section 0 This section causes a 256 word 3X or 9 worst case noise test pattern to be relocated throughout memory 0 = 28K.

NOTE

This should not be constructed to be a memory test.

Section 1 This section tests the unary instruction set executing each unary instruction in each address mode (excluding unary instructions using address mode 7).

Section 2 This section tests the unary instructions using address mode 7 and binaries in all address modes (excluding binary bytes OPS using address mode 7).

Section 3 This section tests binary byte OPS using address mode 7, JMP, JSR and program trap (IOT, TRAP, and EMT) instructions.

Section A Following Section 3 is a routine to ascertain which OP the program is running on. The results are used by the following sections to check the additional instructions/features of the 11/40 and 11/45. This routine leaves the results in location 'OPT,CP'. The contents of this location are typed out as follows:

where:

BIT15 = 1/0 = Memory management option
available/not available
BIT14 = 1/0 = EIS available/not available

NOTE

EIS is always available on
PDP-11/45.

BIT13 = 1/0 = 11/45 FPP available/not
available
BIT12 = 1/0 = 11/40 FIS available/not
available
BIT11 = 1/0 = Stack limit (11/40 K7 option)
available/not available
BIT10 = 1/0 = KW11-P available/not available
BIT09 = 1/0 = KW11-L available/not available
BIT08 = 1/0 = Console TTY available/not
available
BITS 07-00 = 06 = 11/45, 04 = 11/40.

Section 4 This section checks that each bit in the processor
status word (PSW) can be set cleared, reserved
instruction, and odd address traps.
Section 5 This section checks the SXT, XOR, SOB, MARK, RTT
and RTY instructions.
Section 6 This section checks the ASH, ASHC, MUL, DIV, SPL
instructions and the program interrupt request
(PIRQ) logic.
Section 7 This section checks the stack limit register
(KW=11 option on 11/40), and memory management
abort logic (if system has more than 32K of
memory).

Following Section 7 are two routines to check the Teletype printer
logic and a routine to start either the KW11-P or the KW11-L clock.
If either the KW11-P or the KW11-L is available the priority
arbitration logic is tested.

The program then relocates to 160000 (if available) and restarts. The
program continues relocating by increments of 4000 bytes (1K) until
the end of memory is reached. Relocation of the program throughout

all memory constitutes a pass. When the program is executing above 28K, you will hear several 'kerchunks' on the teletype. The 'kerchunks' are caused by the Teletype test following Section 7 mentioned above.

Upon completion of a pass the program restarts using a new processor status depending on the type of processor and the pass count.

7.1 STACK POINTER

The stack Pointer is set at 500.

NOTE

If the program is running in either user or supervisor mode, the user/supervisor stack pointer is set to 500 and the kernel stack pointer is set to 600. The kernel stack pointer is used only for the SCOPE HIT, TTY, and Clock Trap/Interrupt routines.

7.2 POWER FAILURE

A Power Fail service routine is incorporated in the test. When using this program the power should be turned off when running to check the power fail logic. When the power fails the program will type:

POWER FAILED

and restart the program at the beginning (START).

APPENDIX A
KT11 C/D REGISTERS

SR0=777572 = S T A T U S R E G I S T E R #0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NRA	PLE	AV	MMT	PAS	Ø	TE	MM	IC	MO	DE	I/D	PA	GE	NO	ENA

NRA - NON-RESIDENT ABORT	• IC - INSTRUCTION COMPLETED
PLE - PAGE LENGTH ERROR	MODE- CPU MODE
AV - ABORT - READ ONLY ACCESS VIOLATION	ØØ - KERNAL 1Ø - NOT USED
* MMT - MEMORY MANAGEMENT TRAP	Ø1 - SUPERVISOR 11 - USER
* PAS - PROGRAMS AID SYSTEM FLAG	• I/D - SEGMENT ADDRESS SPACE
* TE - ENABLE MEMORY MANAGEMENT TRAPS	Ø - I SPACE 1 -D SPACE
MM - MAINTENANCE MODE	PAGE- PAGE #
	ENA - ENABLE MEMORY MANAGEMENT

* SR1=777574 = S T A T U S R E G I S T E R #1

SR1 RECORDS ANY AUTOINCREMENT/DECREMENT OF THE GPR'S, INCLUDING EXPLICIT REFERENCES THROUGH THE PC, SR1 IS CLEARED AT THE BEGINNING OF EACH INSTRUCTION FETCH, WHENEVER A GPR IS EITHER AUTO INCREMENTED/DECREMENTED THE REGISTER NUMBER AND THE AMOUNT (IN 2'S COMPLEMENT NOTATION) IS RECORDED IN SR1.

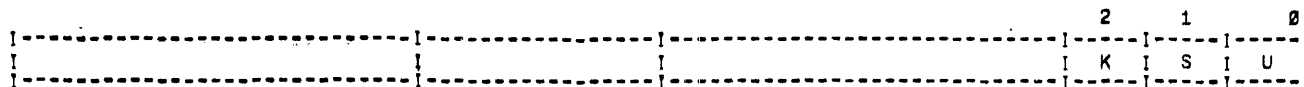
15	11	10	8	7	3	2	0		
AMOUNT				REG NO		AMOUNT		REG NO	

SR2=777576 = S T A T U S R E G I S T E R #2

SR2 IS LOADED WITH THE 16 BIT VIRTUAL ADDRESS AT THE BEGINNING OF EACH INSTRUCTION FETCH, OR WITH THE ADDRESS TRAP VECTOR AT THE BEGINNING OF AN INTERRUPT, (T) BIT TRAP, PARITY, ODD ADDRESS, AND TIMEOUT TRAPS.

• SR3=772516 = S T A T U S R E G I S T E R #3

SR3 ENABLES OR DISABLES THE USE OF I/D SPACE. WHEN 'D' SPACE IS
 DISABLED ALL REFERENCES USE THE 'I' SPACE REGISTERS; WHEN 'D' SPACE
 IS ENABLED BOTH 'I' AND 'D' REGISTERS ARE USED. A '1' ENABLES 'D' SPACE.

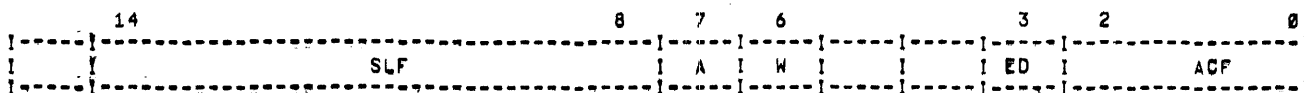


U - USER D SPACE ENABLE
 S - SUPERVISOR 'D' SPACE ENABLE
 K - KERNEL 'D' SPACE ENABLE

THE PROGRAM DOES NOT ENABLE ANY 'D' SPACES

P A G E D E S C R I P T O R R E G I S T E R S

KIPDR0=KIPDR7	772300=772316	*KOPDR0-KOPDR7	772320=772336
UIPDR0=UIPDR7	777600=777616	*UPDR0=UPDR7	777620=777636
*SIPDR0=SIPDR7	772200=772216	*SOPDR0=SOPDR7	772220=772236



ACF - ACCESS CONTROL FIELD	ED - EXPANSION DIRECTION (1/0 = DOWN/UP)
000 NON-RESIDENT	W - SEGMENT HAS BEEN WRITTEN INTO
100 READ/WRITE	* A - SEGMENT HAS BEEN ACCESSED
* 101 READ/WRITE	
110 READ/WRITE	SLF - SEGMENT LENGTH FIELD
* 001 READ ONLY	BINARY REPRESENTATION OF NUMBER OF 32 WORD
010 READ ONLY	BLOCKS IN SEGMENT.
* 011 UNUSED	
* 111 UNUSED	

• 11/45 ONLY

APPENDIX B
RC11 REGISTERS

INTERRUPT VECTOR = 210

RC1A=777440 = L O O K A H E A D R E G I S T E R

```

15          12  11  10          6  5          0
|-----|-----|-----|-----|-----|-----|
| BA |-----| UNIT |-----| TRACK NO |-----| SECTOR |
|-----|-----|-----|-----|-----|-----|

```

RCDA=777442 = D I S K A D D R E S S R E G I S T E R

```

          12  11  10          6  5          0
|-----|-----|-----|-----|-----|-----|
|-----| UNIT |-----| TRACK NO |-----| SECTOR |
|-----|-----|-----|-----|-----|-----|

```

RCER=777444 = D I S K E R R O R R E G I S T E R

```

          9          4
|-----|-----|-----|-----|-----|-----|-----|-----|
| DLE | BCE | DSE | NXM |-----| ATE |-----| APE | SAE | DOE | MXF |-----|-----|
|-----|-----|-----|-----|-----|-----|-----|-----|

```

DLE = DATA LATE ERROR APE = ADDRESS PARITY ERROR
BCE = BLOCK CHECK ERROR SAE = SYNC ADDRESS ERROR
DSE = DATA SYNC ERROR DOE = DISK OVERFLOW ERROR
NXM = NON-EXIST MEMORY MXF = MISSED TRANSFER
ATE = A TRACK ERROR

RCDS=777446 = D I S K C O N T R O L & S T A T U S R E G I S T E R

```

|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| SC | DE | AE | WLE | NXD | WCE | BAI | ABT | RDY | IE | A17 | A16 | MM | FUN | FUN | GO |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|

```

SC = SPECIAL CONDITION A17 = BUS ADDRESS BIT 17
DE = DATA ERROR A16 = BUS ADDRESS BIT 16
AE = ADDRESS ERROR MM = MAINTENANCE MODE
WLE = WRITE LOCK ERROR FUN = FUNCTION
NXD = NON-EXISTENT DISK 00 LOOK AHEAD 10 READ
WCE = WRITE CHECK ERROR 01 WRITE 11 WRITE CHECK
BAI = INHIBIT CA INCREMENT GO = GO
ABT = ABORT
RDY = READY
IE = INTERRUPT ENABLE

RCWC=777450 = W O R D C O J N T R E G I S T E R

RCCA=777452 = C U R R E N T A D D R E S S R E G I S T E R

RCMN=777459 = M A I N T E N A N C E R E G I S T E R

RCDB=777456 = D A T A B U F F E R R E G I S T E R

APPENDIX C
RF11 REGISTERS

INTERRUPT VECTOR # 204

RFDCS=777460 * D I S K C O N T R O L S T A T U S R E G I S T E R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ERR	FRZ	WCE	DPE	NED	WLO	MXF	CLR	RDY	IE	A17	A16	MM	FUN	GO	

ERR	ERR (LOGICAL OR OF 14-9)	IE	INTERRUPT ENABLE
FRZ	FREEZE (LOGICAL OR OF DAE 15-10)	A17	BUS ADDRESS BIT 17
WCE	WRITE CHECK ERROR	A16	BUS ADDRESS BIT 16
DPE	DATA PARITY ERROR	MM	MAINTENANCE MODE
NED	NON EXISTANT DISK	FUN	FUNCTION
WLO	WRITE LOCK OUT		00 NO OP 10 READ
MXF	MISSED TRANSFER		01 WRITE 11 WRITE CHECK
CLR	DISK CLEAR		
RDY	CONTROL READY		

RFWC=777462 * W O R D C O J N T R E G I S T E R

RFDMA=777464 * C U R R E N T M E M O R Y A D D R E S S

RFDAR=777466 * D I S K A D D R E S S R E G I S T E R

15	11	10	0
TRACK		WORD	ADDRESS

RFDAE=777470 * D I S K A D D R E S S E X T E N S I O N E R R O R R E G I S T E R

15	14	13	12	10	8	7	5	4	3	2	1	0
APE	ATE	BTE	CTE	NXM	BAI	DRL	DAO	UNIT	TA			

APE	ADDRESS PARITY ERROR	DRL	DATA REQUEST LATE
ATE	A TIMING TRACK ERROR	DAO	DISK ADDRESS OVERFLOW
BTE	B TIMING TRACK ERROR	UNIT	UNIT NUMBER
CTE	C TIMING TRACK ERROR	TA	EXTENSION OF TRACK ADDRESS
NXM	NON-EXISTANT MEMORY		
BAI	BUS ADDRESS INHIBIT		IN RFDAR

RFDBR=777422 * D A T A B U F F E R R E G I S T E R

RFMA=777474 * M A I N T E N A N C E R E G I S T E R

RFADS=777476 * L O O K A H E A D

10	0
DISK	SEGMENT ADDRESS

APPENDIX D
RK11 REGISTERS

INTERRUPT VECTOR = 220

RK0S=777400 - D R I V E S T A T U S R E G I S T E R

```

I-----I-----I-----I-----I-----I-----I-----I-----I-----I-----I-----I-----I
I   ID       I DPL I RKS I DRV I SIN I SOK I DRY I RDY I WDS I SC=SA I           SECTOR CTR I
I-----I-----I-----I-----I-----I-----I-----I-----I-----I-----I-----I-----I
  
```

ID	IDENT OF DRIVE	DRV	DRIVE READY
DPL	DRIVE POWER LOW	RDY	READ/WRITE/SEEK READY
RKS	SET TO INDICATE RK05	WPS	WRITE PROTECT STATUS
DRV	DRIVE UNSAFE	SC-SA	SECTOR COUNTER=SECTOR ADDRESS
SIN	SEEK INCOMPLETE	SC	SECTOR
SOK	SECTOR COUNTER OK		

RKER=777402 - E R R O R R E G I S T E R

```

I-----I-----I-----I-----I-----I-----I-----I-----I-----I-----I-----I-----I
I DRE I OVR I WLO I SKE I PGE I NXM I DLT I TE I NXD I NXC I NXS I           I           I CSE I WCE I
I-----I-----I-----I-----I-----I-----I-----I-----I-----I-----I-----I-----I
  
```

DRE	DRIVE ERROR	TE	TIMING ERROR
OVR	DISK OVERRUN (OVERFLOW)	NXD	NON-EXISTANT DISK
WLO	WRITE LOCK	NXC	NON-EXISTENT CYLINDER
SKE	SEEK ERROR	NXS	NON-EXISANT SECTOR
PGE	PROGRAMMING ERROR	CSE	CHECK SUM ERROR
NXM	NON-EXISTANT	WCE	WRITE CHECK ERROR
DLT	DATA LATE		

RKCS-777404 - C O N T R O L S T A T U S R E G I S T E R

```

|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| ERR | HE | SCP |   | BAI | FMT |   | SSE | RDY | IE | A17 | A16 |   | FUN |   | GO |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|

```

```

ERR  ERROR(OR OF RKER & RKCS<14>)  A17  BUS ADDRESS BIT 17
HE   HARD ERROR                    A16  BUS ADDRESS BIT 16
SCP  SEARCH COMPLETE                FUN  FUNCTION
BAI  BUS ADDRESS INHIBIT INC.      000  RESET    100  SEEK
FMT  FORMAT MODE                   001  WRITE    101  RD CHK
SSE  STOP ON SOFT ERROR            010  READ     110  DRV RESET
RDY  CONTROL READY                 011  WT CHK   111  WT LOCK
IE   INTERRUPT ENABLE              GO   ENABLE FUNCTION

```

RKWC-777406 - W O R D C O J N T R E G I S T E R

RKLA-777410 - C U R R E N T B U S A D D R E S S

RKDA-777412 - D I S K A D D R E S S

```

|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| DR SEL |   | CYLINDER | SUR |   | SECTOR |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|

```

RKDB-777416 - D A T A B U F F E R R E G I S T E R

APPENDIX E
RP11C REGISTERS

INTERRUPT VECTOR = 254

RPDS-776710 - DRIVE STATUS REGISTER

```

|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| SUR | ONL | RP03 | HNF | SI | SU | UNSF | WP | ATN7 | ATTENTION | ATN0 |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|

```

SUR	SELECTED UNIT READY	SU	SEEK UNDERWAY
ONL	SELECTED UNIT ON LINE	UNSF	SELECTED UNIT UNSAFE
RP03	SELECTED UNIT IS RP03	WLO	SELECTED UNIT WRITE LOCKED
HNF	HEADER NOT FOUND	ATN7-ATN0	DRIVE ATTENTION
SI	SEEK INCOMPLETE		

RPER-776712 - ERROR REGISTER

```

|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| WPV | FUV | NXC | NXT | NXS | PGE | FMT | MOD | LPE | WPE | CSE | TE | WCE | NXM | EOP | DER |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|

```

WPV	WRITE PROT VIOL	LPE	LONG PARITY ERROR
FUV	FILE UNSAFE VIOL	WPE	WORD PARITY ERROR
NXC	NON-EXISTANT CYLINDER	CSE	CHECK SUM ERROR
NXT	NON-EXISTANT TRACK	TE	TIMING ERROR
NXS	NON-EXISTANT SECTOR	WCE	WRITE CHECK ERROR
PGE	PROGRAM ERROR	NXM	NON-EXISTANT MEMORY
FMT	FORMAT ERROR	EOP	END OF PACK
MOD	MODE ERROR	DER	DISK ERROR

RPCS-776714 - CONTROL STATUS REGISTER

```

|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| ERR | HE | AIE | MOD | HDR | DRIVE SELECT | RDY | IE | A17 | A16 | FUNCTION | GO |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|

```

ERR	ERROR (OR OF ALL ERROR BITS)	A17	BUS ADDRESS BIT 17
HE	HARD ERROR (OR OF ALL BUT DATA ERROR BITS)	A16	BUS ADDRESS BIT 16
AIE	ATTENTION INTERRUPT ENABLE	FUN	FUNCTION
MODE	RP11-C IS CONDITIONED TO RD /WRT DISK PACKS IN POP10 OR PDP 15 FORMAT	000	INIT 100 SEEK
		001	WRITE 101 WRITE (NO SEEK)
		010	READ 110 HOME SEEK
		011	WRT CK 111 RD (NO SEEK)
HDR	FUNCTION IS A HEADER OPER,	GO	ENABLE OPERATION
DRIVE SELECT			
RDY	READY		
IE	INTERRUPT ENABLE		

RPWC-776716 - WORD COJNT REGISTER

RPBA-776720 - BUS ADDRESS REGISTER

RPCA-776722 - CYLINDER ADDRESS REGISTER

<BITS 00-08>

RPDA-776724 - D I S K A D D R E S S R E G I S T E R

	TRACK ADDRESS	CURRENT SECTOR	SECTOR ADDRESS
--	---------------	----------------	----------------

SUCA-776734 - S E L E C T E D U N I T C Y L I N D E R A D R S
<BITS 00-08>

RSER-772054 - E R R O R R E G I S T E R

```

I-----I-----I-----I-----I-----I-----I-----I-----I-----I-----I
I DCK I UNS I OPI I DTE I WLE I IAE I AO I I I I I I I I I I I I I I I I I I I I I I I
I-----I-----I-----I-----I-----I-----I-----I-----I-----I-----I

```

DCK	DATA CHECK	AO	DISK ADDRESS OVERFLOW
UNS	DRIVE UNSAFE	PAR	MASSBUS PARITY ERROR
OPI	OPERATION INCOMPLETE	RMR	REGISTER MODIFY REFUSED
DTE	DRIVE TIMING ERROR	ILR	ILLEGAL REGISTER REF.
WLE	WRITE LOCK ERROR	ILF	ILLEGAL FUNCTION
IAE	INVALID DISK ADDRESS		

RSAS-772056 - A T T E N T I O N S U M M A R Y

```

I-----I-----I-----I-----I-----I-----I-----I-----I-----I-----I
I I I I I I I I I I I I I I I I I I I I I I I I I I I I I I I I I I I I I I I I I I
I-----I-----I-----I-----I-----I-----I-----I-----I-----I-----I

```

RSLA-772060 - L O O K A H E A D

```

I-----I-----I-----I-----I-----I-----I-----I-----I-----I-----I
I I I I I I I I I I I I I I I I I I I I I I I I I I I I I I I I I I I I I I I I I I
I-----I-----I-----I-----I-----I-----I-----I-----I-----I-----I

```

RSDB-772062 - D A T A B U F F E R

RSHR-772064 - M A I N T R E G I S T E R

RSDT-772066 - D R I V E T Y P E R E G I S T E R

APPENDIX A
KT11 C/D REGISTERS

SR0=777572 - S T A T U S R E G I S T E R #0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NRA	PLE	AV	MMT	PAS	0	TE	MM	IC	MO	DE	I/D	PA	GE	NO	ENA

NRA - NON-RESIDENT ABORT	* IC - INSTRUCTION COMPLETED
PLE - PAGE LENGTH ERROR	MODE- CPU MODE
AV - ABORT - READ ONLY ACCESS VIOLATION	00 - KERNAL 10 - NOT USED
* MMT - MEMORY MANAGEMENT TRAP	01 - SUPERVISOR 11 - USER
* PAS - PROGRAMS AID SYSTEM FLAG	* I/D - SEGMENT ADDRESS SPACE
* TE - ENABLE MEMORY MANAGEMENT TRAPS	0 - I SPACE 1 -O SPACE
MM - MAINTENANCE MODE	PAGE- PAGE #
	ENA - ENABLE MEMORY MANAGEMENT

* SR1=777574 - S T A T U S R E G I S T E R #1

SR1 RECORDS ANY AUTOINCREMENT/DECREMENT OF THE GPR'S, INCLUDING EXPLICIT REFERENCES THROUGH THE PC. SR1 IS CLEARED AT THE BEGINNING OF EACH INSTRUCTION FETCH, WHENEVER A GPR IS EITHER AUTO INCREMENTED/DECREMENTED THE REGISTER NUMBER AND THE AMOUNT (IN 2'S COMPLEMENT NOTATION) IS RECORDED IN SR1.

15	11	10	8	7	3	2	0
AMOUNT				REG NO	AMOUNT		REG NO

SR2=777576 - S T A T U S R E G I S T E R #2

SR2 IS LOADED WITH THE 16 BIT VIRTUAL ADDRESS AT THE BEGINNING OF EACH INSTRUCTION FETCH, OR WITH THE ADDRESS TRAP VECTOR AT THE BEGINNING OF AN INTERRUPT, 'T' BIT TRAP, PARITY, ODD ADDRESS, AND TIMEOUT TRAPS.

* SR3-772516 = S T A T U S R E G I S T E R #3

SR3 ENABLES OR DISABLES THE USE OF I/O SPACE, WHEN 'D' SPACE IS
 DISABLED ALL REFERENCES USE THE 'I' SPACE REGISTERS; WHEN 'D' SPACE
 IS ENABLED BOTH 'I' AND 'D' REGISTERS ARE USED, A '1' ENABLES 'D' SPACE.

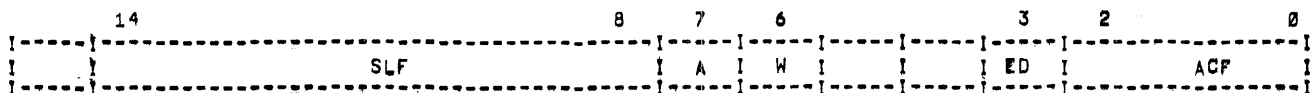


U = USER D SPACE ENABLE
 S = SUPERVISOR 'D' SPACE ENABLE
 K = KERNEL 'D' SPACE ENABLE

THE PROGRAM DOES NOT ENABLE ANY 'D' SPACES

P A G E D E S C R I P T O R R E G I S T E R S

KIPDR0-KIPDR7	772300-772316	*KOPDR0-KOPDR7	772320-772336
UIPDR0-UIPDR7	777600-777616	*UPDPR0-UPDPR7	777620-777636
* SIPDR0-SIPDR7	772200-772216	*SDPDR0-SDPDR7	772220-777236



ACF = ACCESS CONTROL FIELD	ED = EXPANSION DIRECTION (1/0 = DOWN/UP)
000 NON-RESIDENT	W = SEGMENT HAS BEEN WRITTEN INTO
100 READ/WRITE	* A = SEGMENT HAS BEEN ACCESSED
* 101 READ/WRITE	
110 READ/WRITE	SLF = SEGMENT LENGTH FIELD
* 001 READ ONLY	BINARY REPRESENTATION OF NUMBER OF 32 WORD
010 READ ONLY	BLOCKS IN SEGMENT.
* 011 UNUSED	
* 111 UNUSED	

* 11/45 ONLY

APPENDIX B
RC11 REGISTERS

INTERRUPT VECTOR = 210

RCIA=777440 - L O O K A H E A D R E G I S T E R

```

15-----12 11 10-----6 5-----0
|-----|-----|-----|-----|-----|-----|
| BA | | UNIT | | TRACK NO | | SECTOR |
|-----|-----|-----|-----|-----|-----|

```

RCDA=777442 - D I S K A D D R E S S R E G I S T E R

```

-----12 11 10-----6 5-----0
|-----|-----|-----|-----|-----|-----|
| | UNIT | | TRACK NO | | SECTOR |
|-----|-----|-----|-----|-----|-----|

```

RCER=777444 - D I S K E R R O R R E G I S T E R

```

-----9-----4-----
|-----|-----|-----|-----|-----|-----|-----|
| DLE | BCE | DSE | NXM | | ATE | | APE | SAE | DOE | MXF | | | |
|-----|-----|-----|-----|-----|-----|-----|

```

DLE - DATA LATE ERROR APE - ADDRESS PARITY ERROR
BCE - BLOCK CHECK ERROR SAE - SYNC ADDRESS ERROR
DSE - DATA SYNC ERROR DOE - DISK OVERFLOW ERROR
NXM - NON-EXISTENT MEMORY MXF - MISSED TRANSFER
ATE - A TRACK ERROR

RCCS=777446 - D I S K C O N T R O L & S T A T U S R E G I S T E R

```

|-----|-----|-----|-----|-----|-----|-----|-----|
| SC | DE | AE | WLE | NXD | WCE | BAI | ABT | RDY | IE | A17 | A16 | MM | FUN | FUN | GO |
|-----|-----|-----|-----|-----|-----|-----|-----|

```

SC - SPECIAL CONDITION A17 - BUS ADDRESS BIT 17
DE - DATA ERROR A16 - BUS ADDRESS BIT 16
AE - ADDRESS ERROR MM - MAINTENANCE MODE
WLE - WRITE LOCK ERROR FUN - FUNCTION
NXD - NON-EXISTENT DISK 00 LOOK AHEAD 10 READ
WCE - WRITE CHECK ERROR 01 WRITE 11 WRITE CHECK
BAI - INHIBIT CA INCREMENT GO - GO
ABT - ABORT
RDY - READY
IE - INTERRUPT ENABLE

RCWC=777450 - W O R D C O J N T R E G I S T E R

RCCA=777452 - C U R R E N T A D D R E S S R E G I S T E R

RCMN=777459 - M A I N T E N A N C E R E G I S T E R

RCDB=777456 - D A T A B U F F E R R E G I S T E R

APPENDIX D
RK11 REGISTERS

INTERRUPT VECTOR = 220

RKDS=777400 - D R I V E S T A T U S R E G I S T E R

```

|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| ID      | DPL | RKS | DRV | SIN | SOK | DRY | RDY | WDS | SC=SA | SECTOR CTR |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|

```

```

ID      IDENT OF DRIVE      DRY      DRIVE READY
DPL     DRIVE POWER LOW  RDY      READ/WRITE/SEEK READY
RKS     SET TO INDICATE RK05  WPS     WRITE PROTECT STATUS
DRV     DRIVE UNSAFE      SC=SA   SECTOR COUNTER=SECTOR ADDRESS
SIN     SEEK INCOMPLETE   SC      SECTOR
SOK     SECTOR COUNTER OK

```

RKER=777402 - E R R O R R E G I S T E R

```

|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| DRE | OVR | WLO | SKE | PGE | NXM | DLT | TE | NXD | NXC | NXS |   |   | CSE | WCE |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|

```

```

DRE     DRIVE ERROR
OVR     DISK OVERRUN (OVERFLOW)
WLO     WRITE LOCK
SKE     SEEK ERROR
PGE     PROGRAMMING ERROR
NXM     NON-EXISTANT
DLT     DATA LATE
TE      TIMING ERROR
NXD     NON-EXISTANT DISK
NXC     NON-EXISTENT CYLINDER
NXS     NON-EXISANT SECTOR
CSE     CHECK SUM ERROR
WCE     WRITE CHECK ERROR

```

RKGS-777404 - C O N T R O L S T A T U S R E G I S T E R

```

|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| ERR | HE | SCP |   | BAI | FMT |   | SSE | RDY | IE | A17 | A16 |   | FUN |   | GO |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|

```

```

ERR  ERROR(OR OF RKER & RKCS<14>)  A17  BUS ADDRESS BIT 17
HE   HARD ERROR                     A16  BUS ADDRESS BIT 16
SCP  SEARCH COMPLETE                 FUN  FUNCTION
BAI  BUS ADDRESS INHIBIT INC.        000  RESET    100  SEEK
FMT  FORMAT MODE                    001  WRITE   101  RD CHK
SSE  STOP ON SOFT ERROR              010  READ   110  DRV RESET
RDY  CONTROL READY                   011  WT CHK  111  WT LOCK
IE   INTERRUPT ENABLE                GO   ENABLE FUNCTION

```

RKWC-777406 - W O R D C O J N T R E G I S T E R

RKLA-777410 - C U R R E N T B U S A D D R E S S

RKDA-777412 - D I S K A D D R E S S

```

|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| DR SEL | I |           CYLINDER           | SUR |           SECTOR           |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|

```

RKDB-777416 - D A T A B U F F E R R E G I S T E R

APPENDIX E
RP11C REGISTERS

INTERRUPT VECTOR = 254

RPDS-776710 - D R I V E S T A T U S R E G I S T E R

```

|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| SUR | ONL | RP03 | HNF | SI | SU | UNSF | WP | ATN7 | ATTENTION | ATN0 |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|

```

SUR	SELECTED UNIT READY	SU	SEEK UNDERWAY
ONL	SELECTED UNIT ON LINE	UNSF	SELECTED UNIT UNSAFE
RP03	SELECTED UNIT IS RP03	WLO	SELECTED UNIT WRITE LOCKED
HNF	HEADER NOT FOUND	ATN7=ATN0	DRIVE ATTENTION
SI	SEEK INCOMPLETE		

RPER-776712 - E R R O R R E G I S T E R

```

|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| WPV | FUV | NXC | NXT | NXS | PGE | FMT | MOD | LPE | WPE | CSE | TE | WCE | NXM | EOP | DER |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|

```

WPV	WRITE PROT VIOL	LPE	LONG PARITY ERROR
FUV	FILE UNSAFE VIOL	WPE	WORD PARITY ERROR
NXC	NON-EXISTANT CYLINDER	CSE	CHECK SUM ERROR
NXT	NON-EXISTANT TRACK	TE	TIMING ERROR
NXC	NON-EXISTANT SECTOR	WCE	WRITE CHECK ERROR
PGE	PROGRAM ERROR	NXM	NON-EXISTANT MEMORY
FMT	FORMAT ERROR	EOP	END OF PACK
MOD	MODE ERROR	DER	DISK ERROR

RPCS-776714 - C O N T R O L S T A T U S R E G I S T E R

```

|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| ERR | HE | AIE | MOD | HDR | DRIVE SELECT | RDY | IE | A17 | A16 | FUNCTION | GO |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|

```

ERR	ERROR (OR OF ALL ERROR BITS)	A17	BUS ADDRESS BIT 17
HE	HARD ERROR (OR OF ALL BUT DATA ERROR BITS)	A16	BUS ADDRESS BIT 16
AIE	ATTENTION INTERRUPT ENABLE	FUN	FUNCTION
MODE	RP11-C IS CONDITIONED TO RD /WRT DISK PACKS IN POP10 OR PDP 15 FORMAT	000	INIT 100 SEEK
		001	WRITE 101 WRITE (NO SEEK)
		010	READ 110 HOME SEEK
		011	WRT CK 111 RD (NO SEEK)
HDR	FUNCTION IS A HEADER OPER,	GO	ENABLE OPERATION
DRIVE SELECT			
RDY	READY		
IE	INTERRUPT ENABLE		

RPWC-776716 - W O R D C O J N T R E G I S T E R

RPBA-776720 - B U S A D D R E S S R E G I S T E R

RPCA-776722 - C Y L I N D E R A D D R E S S R E G I S T E R
<BITS 00=08>

RPDA-776724 - D I S K A D D R E S S R E G I S T E R

	TRACK ADDRESS	CURRENT SECTOR	SECTOR ADDRESS
--	---------------	----------------	----------------

SUCA-776734 - S E L E C T E D U N I T C Y L I N D E R A D R S
<BITS 00-00>

RSER-772054 - E R R O R R E G I S T E R

```

I-----I-----I-----I-----I-----I-----I-----I-----I
I DCK I UNS I OPI I DTE I WLE I IAE I  AO I           I          I PAR I RMR I ILR I ILF I
I-----I-----I-----I-----I-----I-----I-----I-----I

```

DCK	DATA CHECK	AO	DISK ADDRESS OVERFLOW
UNS	DRIVE UNSAFE	PAR	MASSBUS PARITY ERROR
OPI	OPERATION INCOMPLETE	RMR	REGISTER MODIFY REFUSED
DTE	DRIVE TIMING ERROR	ILR	ILLEGAL REGISTER REF.
WLE	WRITE LOCK ERROR	ILF	ILLEGAL FUNCTION
IAE	INVALID DISK ADDRESS		

RSAS-772056 - A T T E N T I O N S U M M A R Y

```

I-----I-----I-----I-----I-----I-----I-----I-----I
I           I           I           I           I           I           I           I
I           I AT7 I AT6 I AT5 I AT4 I AT3 I AT2 I AT1 I AT0 I
I-----I-----I-----I-----I-----I-----I-----I-----I

```

RSLA-772060 - L O O K A H E A D

```

I-----I-----I-----I-----I-----I-----I-----I-----I
I           I           I           I           I           I           I           I
I           I           I CURRENT ADDRESS I           I SECTOR FRACTION I
I-----I-----I-----I-----I-----I-----I-----I-----I

```

RSDB-772062 - D A T A B U F F E R

RSMR-772064 - M A I N T R E G I S T E R

RSDT-772066 - D R I V E T Y P E R E G I S T E R

DCQKCD 11/40-11/45 CPU EXERCISER
DCQKCD TABLE OF CONTENTS

MACY11 27(655) 4-SEP-74 11153

5	SWITCH SETTING
36	DEFINITIONS & ASSIGNMENTS
285	ENABLE PARITY & POWER FAIL ROUTINES
339	PROG INDICATORS & SCOPE ROUTINE
416	RELOC ROUTINE
475	IODEV ROUTINE
661	DEVICE TABLES
762	TYPE SUBROUTINE
907	ERROR SERVICE ROUTINE
1118	PARITY ERROR SERVICE
1219	MISC SUBROUTINES
1257	KT ABORT, RESERVED & ERROR TRAP SERVICE
1299	PROGRAM INITIALIZATION
1426	START OF SECTION 0
1534	START OF SECTION 1
2671	START OF SECTION 2
3692	START OF SECTION 3
4201	START OF SECTION 4
4545	START OF SECTION 5
4955	START OF SECTION 6
5368	START OF SECTION 7
5604	TELETYPE & CLOCK TESTS
5842	STMM ROUTINE

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54

```

INLIST MD,MC
LIST ME
ABS
TITLE DCQKCD 11/40-11/45 CPU EXERCISER
$BTTL SWITCH SETTING
SW15---HALT ON ERROR
SW14---LOOP TEST
SW13---INHIBIT ERROR TYPEOUT
SW12---SEE NOTE BELOW
SW11---INHIBIT TEST ITERATIONS
SW10---RING BELL ON ERROR
SW09---SEE NOTE BELOW
SW08---LOAD MICRO BRAK REGISTER WITH SW07-SW00
SW07---TYPE END OF PASS MESSAGE
SW06---DISABLE CLOCKS
SW05---RELOCATE USING ALL DEVICES ROUND ROBIN STYLE
SW04---USE RANDOM DISK ADDRESS FOR RELOCATION
SW03---RELOCATE USING DEVICE SELECTED IN SW02-SW00
SW02=SW00--- 0=CP
                1=RK
                2=RF
                3=RP
                4=RC
                5=DO NOT USE
                6=RS03/RS04
                7=RESERVED FOR FUTURE USE (IS CP)
NOTE BELOW SW12 AND SW09 CONTROL PROGRAM RELOCATION DESCRIBED BELOW
SW12 SW09 RELOCATION
1 0 NONE
0 1 NO RELOCATION ABOVE 28K
1 1 NOT USED (DO NOT USE)
0 0 ALL MEMORY

```

\$BTTL DEFINITIONS & ASSIGNMENTS

GENERAL REGISTER ASSIGNMENTS

```

R0=X0
R1=X1
R2=X2
R3=X3
R4=X4
R5=X5
SP=X6
PC=X7
R10=X0
R11=X1
R12=X2
R13=X3
R14=X4
R15=X5

```

FLOATING POINT REGISTERS

AC0=X0

55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108

```

AC1=X1
AC2=X2
AC3=X3
AC4=X4
AC5=X5

ISTACK POINTER REGISTERS
KSP=X6      IKERNEL STACK POINTER
SSP=X6      ISUPERVISOR STACK POINTER
USP=X6      IUSER STACK POINTER

ISTATUS REGISTER (PQW) BIT ASSIGNMENTS
C=1         IC BIT
V=2         IV BIT
Z=4         IZ BIT
N=10        IN BIT
T=20        IT BIT
PRTY7=340  IPRIORITY LEVEL 7
PRTY6=300  IPRIORITY LEVEL 6
PRTY5=240  IPRIORITY LEVEL 5
PRTY4=200  IPRIORITY LEVEL 4
PRTY3=140
PRTY2=100  IPRIORITY LEVEL 2
KM=000000 IKERNEL MODE
SM=040000 ISUPERVISORY MODE
UM=140000 IUSER MODE
PKM=000000 IPREVIOUS KERNEL MODE
PSM=010000 IPREVIOUS SUPERVISORY MODE
PUM=030000 IPREVIOUS USER MODE
REG=004000 ISELECT R10-R15

IVECTOR ADDRESSES
ERRVEC= 4    IADDRESS OF ERROR VECTOR
RESVEC= 10   IADDRESS OF RESERVED INST, TRAP VECTOR
TBITVEC=14   IADDRESS OF 'T' BIT TRAP VECTOR
TRTVEC= 14   IADDRESS OF 'TRACE' TRAP VECTOR
BPTVEC= 14   IADDRESS OF 'BREAKPOINT' TRAP VECTOR
IOTVEC= 20   IADDRESS OF IOT TRAP VECTOR
PFEVC= 24    IADDRESS OF POWER FAIL TRAP VECTOR
EMTVEC= 30   IADDRESS OF EMT VECTOR
TRAPVEC=34   IADDRESS OF TRAP VECTOR
TKVEC= 00    IADDRESS OF KEYBOARD INTERRUPT VECTOR
TPVEC= 64    IADDRESS OF TTY PRINTER INTERRUPT VECTOR
PRVEC= 70    IHIGH SPEED READER INTERRUPT VECTOR
PPVEC= 74    IHIGH SPEED PUNCH INTERRUPT VECTOR
LKVEC= 100   IADDRESS KW11=L LINE CLOCK INT. VECTOR
PLKVEC= 104  IADDRESS OF KW11=P CLOCK INT VECTOR
RFVEC= 204   IRF OR RS04 VECTOR
RSVEC= 204
RCVEC= 210   IRC VECTOR
RKVEC= 220   IRK DISK VECTOR
PIRVEC= 240  IADDRESS OF PIRQ VECTOR
FPEVEC= 244  IADDRESS OF FLOATING POINT INT. VECTOR
MHVEC= 250   IADDRESS OF MEM MGMT ERROR TRAP VECTOR

```

109	000254	RPVEC= 254	JRP VECTOR
110	000254	RP4VEC= 254	JRP04 VECTOR
111			
112		JREGISTER ADDRESSES	
113	177776	PSW= 177776	JADDRESS OF STATUS REGISTER
114	177774	SLR= 177974	JADDRESS OF STACK LIMIT REGISTER
115	177772	PIRQ= 177772	JADDRESS OF PROGRAM INTERRUPT REQUEST
116	177770	UBREAK= 177770	JADDRESS OF MICRO BREAK REGISTER
117	177766	CPUERR= 177966	
118	177744	ERRREG= 177844	
119	177546	LKS= 177546	JADDRESS OF KW11-L STATUS REG,
120	177550	PR0= 177950	JADDRESS OF HIGH SPEED READER CSR
121	177552	PR0= 177952	JADDRESS OF HIGH SPEED READER DATA BUF
122	177594	PPS= 177954	JADDRESS OF HIGH SPEED PUNCH CSR
123	177556	PP0= 177956	JADDRESS OF HIGH SPEED PUNCH BUFFER
124	177560	TKS= 177560	JADDRESS OF KEYBOARD CSR
125	177562	TK0= 177562	JADDRESS OF KEYBOARD BUFFER
126	177564	TPS= 177964	JADDRESS OF TELEPRINTER CSR
127	177566	TP0= 177566	JADDRESS OF TELEPRINTER BUFFER
128	177572	SR0= 177972	JADDRESS OF MEM MGMT REGISTER SR0
129	177574	SR1= 177974	JADDRESS OF MEM MGMT REG SR1
130	177576	SR2= 177976	JADDRESS OF MEM MGMT REGISTER SR2
131	172516	SR3= 172516	JADDRESS OF MEM MGMT REGISTER SR3
132	177570	SHR= 177970	JADDRESS OF CONSOL SWITCH REGISTER
133	177570	DISPLAY= 177570	JADDRESS OF CONSOL DISPLAY REGISTER
134	177514	LPS= 177914	JADDRESS OF LINE PRINTER STATUS REG
135	177516	LP0= 177916	JADDRESS OF LINE PRINTER DATA DUFFER
136			
137		JRK REGISTERS	
138	177400	RKDS= 177400	JADDRESS OF RK=11 DISK DRIVE STATUS REGISTER
139	177402	RKER= 177402	JADDRESS OF RK=11 DISK ERROR REGISTER
140	177404	RKCS= 177404	JADDRESS OF RK=11 DISK CONT, AND STATUS REG,
141	177406	RKWC= 177406	JADDRESS OF RK=11 DISK WORC COUNT REG,
142	177410	RKBA= 177410	JADDRESS OF RK=11 DISK BUS ADDRESS REG,
143	177412	RKDA= 177412	JADDRESS OF RK=11 DISK ADDRESS REG,
144			
145		JRF REGISTERS	
146	177460	RFOCS= 177460	JADDRESS OF RF=11 DISK CONT, AND STATUS REG,
147	177462	RFWC= 177462	JADDRESS OF RF=11 DISK WORC COUNT REG,
148	177464	RFCMA= 177464	JADDRESS OF RF=11 DISK MEMCRY ADR REG,
149	177466	RFDAR= 177466	JADDRESS OF RF=11 DISK ADDRESS REG,
150	177470	RFDAE= 177470	JADDRESS OF RF DAE REGISTER
151			
152		JRC REGISTERS	
153	177440	RCLA= 177440	JADDRESS OF RC=11 LOOK AHEAD REGISTER
154	177442	RCDAR= 177442	JADDRESS OF RC=11 DISK ADDRESS REG,
155	177446	RCCS= 177446	JADDRESS OF RC=11 DISK CONT, AND STATUS REG,
156	177450	RCWC= 177450	JADDRESS OF RC=11 DISK WORC COUNT REG,
157	177452	RCCA= 177452	JADDRESS OF RC=11 CURRENT DISK ADR REG,
158			
159		JRP04 REGISTERS	
160	176700	RP4CS1= 176700	JRP04 CS1 REGISTER
161	176702	RP4WC= 176702	JWORD COUNT REGISTER
162	176704	RP4BA= 176704	JBUS ADDRESS REGISTER

163	176706	RP4ST= 176706	JDESIRED SECTOR/TRACK REGISTER
164	176734	RP4CA= 176734	JDISK ADDRESS REGISTER
165	176712	RP4S1= 176712	JDRIVE STATUS REGISTER #1
166	176714	RP4E1= 176714	JERRON REGISTER #1
167	176716	RP4AS= 176716	JATTENTION SUMMARY
168	176720	RP4LA= 176720	JLOOK AHEAD REGISTER
169	176732	RP4OF= 176732	JOFFSET REGISTER
170			
171		JRM11 MASS BUS CONTROLLER REGISTERS	
172	000000	RMC52= 0	JNOT DEFINED
173			
174		JRP16 REGISTERS	
175	176710	RPDS= 176710	JADDRESS OF RP DRIVE STATUS REGISTER
176	176712	RPER= 176712	JADDRESS OF RP ERROR REGISTER
177	176714	RP0= 176714	JADDRESS OF RP CONTROL STATUS REGISTER
178	176716	RPWC= 176716	JADDRESS OF RP WORD COUNT REGISTER
179	176720	RPBA= 176720	JADDRESS OF RP BUS ADDRESS REGISTER
180	176722	RPCA= 176722	JADDRESS OF RP CYLINDER ADDRESS REGISTER
181	176724	RPDA= 176724	JADDRESS OF RP DISK ADDRESS REGISTER
182			
183		JKW11-P REGISTERS	
184	172540	PLKCSR= 172540	JADDRESS OF KW11-P CLOCK CSR
185	172542	PLKCSB= 172542	JADDRESS OF KW11-P COUNT SET BUFFER
186	172544	PLKCTR= 172544	JADDRESS OF KW11-P COUNTER
187			
188		JRS04 REGISTERS	
189	172040	RS0S1= 172040	JCONTROL STATUS REGISTER
190	172042	RSWC= 172042	JWORD COUNT REGISTER
191	172044	RSBA= 172044	JBUS ADDRESS REGISTER
192	172046	RSDA= 172046	JDISK ADDRESS REGISTER
193	172050	RS0S2= 172050	JCONTROL STATUS #2
194	172052	RS0S= 172052	JDRIVE STATUS REGISTER
195	172054	RSER= 172054	JERROR REGISTER #1
196	172056	RSAS= 172056	JATTENTION SUMMARY REGISTER
197	172060	RSLA= 172060	JLOOK AHEAD REGISTER
198			
199		JMEMORY MANAGEMENT REGISTER ADDRESSES	
200	172300	KIPDR0= 172300	
201	172302	KIPDR1= 172302	
202	172304	KIPDR2= 172304	
203	172306	KIPDR3= 172306	
204	172310	KIPDR4= 172310	
205	172316	KIPDR7= 172316	
206	172340	KIPAR0= 172340	
207	172342	KIPAR1= 172342	
208	172344	KIPAR2= 172344	
209	172346	KIPAR3= 172346	
210	172350	KIPAR4= 172350	
211	172356	KIPAR7= 172356	
212			
213	177600	UIPDR0= 177600	
214	177602	UIPDR1= 177602	
215	177610	UIPDR4= 177610	
216	177614	UIPDR6= 177614	

217 177616
 218 177640
 219 177642
 220 177650
 221 177654
 222 177656
 223
 224 172200
 225 172202
 226 172210
 227 172214
 228 172216
 229 172240
 230 172242
 231 172250
 232 172254
 233 172256
 234 172320
 235 177620
 236 172220
 237 172360
 238 177660
 239 172260
 240
 241
 242 000500
 243 000600
 244
 245
 246
 247 100000
 248 040000
 249 020000
 250 010000
 251 004000
 252 002000
 253 001000
 254 000400
 255
 256
 257 000001
 258 000002
 259
 260 100000
 261 040000
 262 020000
 263 000400
 264 000100
 265 010000
 266
 267
 268 104400
 269 104000
 270 000004

UIPDR7=177616
 UIPAR0=177640
 UIPAR1=177642
 UIPAR4=177650
 UIPAR6=177654
 UIPAR7=177656
 SIPDR0=172200
 SIPDR1=172202
 SIPDR4=172210
 SIPDR6=172214
 SIPDR7=172216
 SIPAR0=172240
 SIPAR1=172242
 SIPAR4=172250
 SIPAR6=172254
 SIPAR7=172256
 WOPDR0=172320
 UOPDR0=177620
 SOPDR0=172220
 WOPAR0=172360
 UOPAR0=177660
 SOPAR0=172260

INITIAL STACK POINTER SETTING

STKPTR= 500 ;PROGRAM STACK PTR
 KPTR= 600 ;KERNEL STACK PTR (USED BY KERNEL WHEN
 ;PROG IS RUNNING IN OTHER THAN KERNEL MODE

MISCELLANEOUS BIT ASSIGNMENTS (USED IN OPT,CP)

KTOPT= 100000 ;BELOW BIT ASSIGNMENTS ARE USED
 EISOPT= 040000 ;IN THE CPCHK ROUTINE
 PPOPT= 020000 ;A BIT FOR EACH OPTION PRESENT
 FISOPT= 010000 ;IS SET IN OPT,CP (ODD BYTE)
 KJOPT= 004000
 PLKOPT= 002000
 LKOPT= 001000
 TTOPT= 000400

BIT ASSIGNMENTS USED IN OPTIONS

PROPT= 000001
 PPOPT= 000002
 BIT15= 100000
 BIT14= 400000
 BIT13= 200000
 BIT8= 400
 BIT6= 100
 PIR4= 10000 ;LEVEL 4 PROGRAM INT. ROST, (FOR PIRD)

INSTRUCTION EQUATES

HLT= TRAP ;HLT IS A TRAP INST TO THE ERROR ROUTINE
 SCOPE= EMT ;SCOPE IS AN EMT TRAP
 TYPE= IOT

```

272          000020          ;=IOTVEC
273 000020 002564          ;WORD TYPE ;SET IOT VECTOR TO TYPE ROUTINE
274 000022 000200          ;WORD PRTY4
275 000024 000610          ;WORD PDWN ;SET POWER FAIL VECTOR
276 000026 000340          ;WORD PRTY7
277 000030 001044          ;WORD SCOPEA
278 000032 000200          ;WORD PRTY4
279 000034 003212          ;WORD HLT
280 000036 000340          ;WORD PRTY7
281          000060          ;=TKVEC
282 000060 003100          ;WORD TKISR ;SET KEYBOARD VECTOR TO TKISR
283 000062 000200          ;WORD PRTY4 ;PRIORITY LEVEL 4
284
285          ;SBTTL ENABLE PARITY & POWER FAIL ROUTINES
286          000120          ;=I20
287          ;ROUTINE TO SET PARITY ACTION ON PARITY MEMORIES
288          PARCSR= 172100 ;ADDRESS OF FIRST POSSIBLE PARITY REG
289          PARVEC= 000114 ;ADDRESS OF PARITY INTERRUPT VECTOR
290
291 000120 012737 004356 000114 ;MAMF MOV #,PARSR,#PARVEC ;LOAD VECTOR
292 000126 012737 000340 000116 ;MOV #340,#PARVEC+2 ;AND PRIORITY LEVEL
293 000134 012737 000006 000004 ;MOV #ERRVEC+2,#ERRVEC ;ODD RTI ON TIME OUT TRAP
294 000142 012700 172100 ;MOV #PARCSR,R0 ;GET FIRST POSSIBLE ADDRESS
295 000146 012702 000001 ;MOV #1,R2 ;SET REGISTER COUNTER
296
297 000152 012720 000001 ;SI MOV #1,(R0)+ ;SET ACTION ENABLE (IF AVAIL)
298 ;ABOVE INSTRUCTION WILL SET ACTION ENABLE IF MA/MF PARITY OR SET
299 ;ODD PARITY AND HALT ON PARITY ERROR IF MOS PARITY
300 000156 006302          ;ASL R2 ;CHECK IF 16, REGISTERS HAVE
301 000160 103374          ;RCC 16 ;BEEN ENABLED
302 000162 000207          ;2SI RTS PC ;RETURN
303
304          ;=200
305 000200 012707 005422          ;MOV #START,PC ;GO TO START OF TEST
306 000204 012707 005532          ;MOV #START1,PC ;GO GET LOWER/UPPER RELOCATION BOUNDARY
307 000210 012707 005600          ;MOV #START3,PC ;START WITH LAST TYPED BOUNDARY LIMITS
308
309          ;=244
310 000244 000246          ;WORD 246 ;SET FIS TRAP TO RETURN DIRECT
311 000246 000002          ;WORD RTI
312          ;=252
313 000252 000340          ;WORD 340 ;MEM MGMT PRIORITY ADDRESS
314          ;=610
315          ;POWER FAIL SUBROUTINE
316 000610 005737 000764 ;PDWN: TST #PORT,CP
317 000614 100002          ;BPL 16
318 000616 005037 177572 ;CLR #MSR0
319 000622 012737 000632 000024 ;SI MOV #PUP,#PFVEC
320 000630 000000          ;HALT
321
322          ;POWER UP SUBROUTINE
323 000632 012737 000610 000024 ;PUP: MOV #PDWN,#PFVEC ;RESET POWER FAIL TRAP VECTOR TO POWER
324          ;DOWN ROUTINE ABOVE
325 000640 012706 000600          ;MOV #KPTR,SP ;SET STACK PTR
    
```

```

326 000644 005027          CLR (PC)+
327 000646 000000          ;SI WORD 0 ;KILL TIME
328 000650 005267 177772          ;2SI INC 16
329 000654 001375          BNE 25
330 000656 000004 000666          TYPE,PFAIL
331 000662 000157 005422          JMP #START ;RESTART TEST
332
333 000666 005015 047520 042527 ;PFAIL .ASCIZ '<15><12>'POWER FAILED'<15><12>'
334 000674 020122 040506 046111
335 000702 042105 005015 000
336 000707 015 050012 051101 ;PARERRI .ASCIZ '<15><12>'PARITY ERROR'<15><12>'
337 000714 052111 020131 051105
338 000722 047522 006522 000012
    
```

```

339          ;SBTTL  PROG INDICATORS & SCOPE ROUTINE
340          ;THE BELOW TABLE CONTAINS ERROR INFORMATION NEEDED TO REPORT
341          ;MEMORY ERRORS DETECTED DURING PROGRAM RELOCATION, THE ERROR INFOR-
342          ;MATION IS PLACED IN THE TABLE BY THE 'SAVVAL' SUBROUTINE, AND
343          ;IS PROCESSED BY THE 'PRINTPE' SUBROUTINE.
344          MEMTBL  .WORD  0          ;CONTAINS 'GOOD' ADDRESS
345          .WORD  0          ;CONTAINS 'GOOD' DATA
346          .WORD  0          ;CONTAINS 'BAD' DATA
347          .WORD  0          ;CONTAINS 'BAD' DATA
348          .WORD  0          ;CONTAINS 'BAD' DATA
349          ECHOI   .WORD  0          ;LOCATION FOR ECHOED CHARACTER
350          DEVERRI .ASCII  ' ERROR'
351          .WORD  0
352          CRLF   .ASCII <15><12>
353          SLASH  .ASCII '\ '
354          DEVI   .BYTE  0          ;CONTAINS DEVICE ID FOR ALL
355          IORETRY .BYTE  0          ;CONTAINS DEVICE RETRY COUNT
356          PEFLG  .BYTE  0          ;PARITY ERROR FLAG
357          .EVEN
358          EABITS .WORD  2          ;CONTAINS EA BITS FOR DISK XFERS
359          OPT,CP .WORD  400        ;CONTAINS OPTION AND CP INDICATORS
360          .EVEN BYTE:14=11/40,6=11/45
361          .ODD BYTE: 200=KT,100=EIS
362          .10=11/45 FPP,20=11/40 FIS
363          .10=STACK LIMIT (KJ)
364          .4=KW11-P;2=KW11-L;1=CONSOLE TTY
365          OPTIONS .WORD  0
366          PRDAT  .BYTE  1          ;CONTAINS NEXT DATA TO BE READ
367          PRSYNCI .BYTE  0          ;CONTAINS SYNC COUNT
368          .=770
369          HMONI  .BYTE  0          ;MEM MGMT ON/OFF IND 1/0=ON/OFF
370          QVI   .BYTE  0          ;QUICK VERIFY MODE IND
371          DEVID  .WORD  0          ;CONTAINS DEVICE IDENTIFIER
372          LTICKS .WORD  0          ;CONTAINS L CLOCK TICK COUNT
373          PTICKS .WORD  0          ;CONTAINS P CLOCK TICK COUNT
374          SGNTL  .WORD  0          ;CONTAINS PASS COUNT
375          SFILLS .WORD  1000       ;CONTAINS FILLS COUNT (2) IN ODD BYTE
376          ;AND FILLER CHARACTER (0) IN EVEN BYTE
377          ;FILLER COUNTS: VT05 #2400 BD#4,VT09 #1200 BD#2,VT05 #600 BD#1
378          ;LA30S #110 BD#2,LA30S #150 BD#4,LA30S #300 BD#12
379          ;ALL VALUES ARE OCTAL
380
381          FACTOR .WORD  0          ;CONTAINS RELOCATION FACTOR, SUBTRACT # IN
382          ;FACTOR FROM PC TO GET PC OF ORIG CODE
383          RELR1  .WORD  0          ;CONTAINS RELOCATED R1 (THE R1 OF THE
384          ;ORIGINAL CODE MOVED)
385          FRSTADI .WORD  0          ;CONTAINS FIRST ADRS OF CODE TO BE MOVED
386          FRSTMEMI .WORD  0          ;CONTAINS LOWER RELOCATION BOUNDARY ADDRESS
387
388          ;SCOPE (EHT) SERVICE ROUTINE
389          ;THIS ROUTINE ALLOWS THE SUBTEST TO BE CONTINUOUSLY LOOPEO, ITERATED
390          ;(OR NOT ITERATED) BEFORE BEGINNING NEXT SUBTEST
391          SCOPEAI .CMB  #10, #OPT,CP
392          .BNE 105
  
```

```

393          CLR    #0CUERR
394          MOV    #1, #ERRREG
395          BIT    #4000, 2(SP) ;WAS REGISTER SET BIT SET
396          BEQ   15          ;BRANCH IF NOT
397          BIR   #4000, #PSW ;RETAIN REGISTER SET
398          BIT   #4000, #PSW ;CHECK BIT 14 (CONTINUOUS LOOP)
399          BEQ   45
400          MOV   R1, (SP)   ;LOAD RETURN ADDRESS
401          MOV   R1, #BELR1
402          SUB   #PCYDR, #RELR1 ;RELR1 CONTAINS UNRELOCATED R1
403          BIT   #400, #SWR ;LOAD PDF11/45 MICRO BREAK REG?
404          BEQ   35
405          MOV  #SWR, #SUBBREAK ;LOAD MICRO BREAK REG WITH SR0-7
406          RTI  ;RETURN TO SUBTEST
407          BIT  #4000, #PSW ;SUBTEST ITERATION DESIRED?
408          BNE  75          ;BRANCH IF NO ITERATION DESIRED?
409          DEC  (PC)+       ;DECREMENT SUBTEST ITERATION COUNT
410          DEC  ;CONTAINS SUBTEST ITERATION COUNT
411          BNE  25
412          MOV  #ITCNT, 55 ;RESET ITERATION COUNT
413          MOV  (SP), R1   ;GET ADDRESS OF NEXT TEST
414          BR   25
415          ITCNTI .WORD  40
  
```



```

416          .SBTTL RELOC ROUTINE
417          ROUTINE TO RELOCATE PROGRAM CODE
418 RELOC: BIT #10000,#SWR I BRANCH TO DW12=0
419          REQ 205 ISN:=1 & SW09=0 = NO RELOCATION
420          RIT #1000,#SWR I BRANCH IF SW09=0
421          REQ 45 I NO RELOCATION IF S12=1 & SW09=0
422          TSTB @#MNON I BRANCH IF MEM MGMT IS ENABLED
423          BNE 45 I NO RELOCATION IF MEM MGMT IS ON
424          MOV @#FRSTAD,R0 I GET FIRST ADDRESS OF CODE TO BE MOVED
425          MOV R0,R5 I SAVE
426          MOV R2,R4 I GET LAST ADDRESS OF CODE TO BE MOVED
427          SUB R5,R4 I R4 CONTAINS # OF BYTES TO RELOCATE
428          MOV R2,R3 I SAVE LAST ADDRESS OF CODE TO BE MOVED
429          TST @#FACTOR I FIRST RELOCATION IS TO ENDTAG+2
430          BNE 105
431          MOV R2,@#RETPC I SAVE RETURN PC TO NEXT SECTION OF CODE
432          MOV @#FRSTMEM,R2 I SET FIRST ADDRESS
433          ADD R2,R4 I R4 CONTAINS LAST MEMORY ADDRESS
434          CMP R4,@#LSTMEM I EXIT IF INSUFFICIENT MEMORY
435          RMI 55 I AVAILABLE FOR RELOCATION
436          SUB R2,R4 I R4 NOW CONTAINS BYTE COUNT
437          CLR @#FACTOR I CLEAR RELOCATION FACTOR
438          BIT #40,#SWR I CHECK IF ALL DEVICES DESIRED FOR
439          RNE 125 I RELOCATION ROUND ROBIN STYLE
440          BIT #10,#SWR I CHECK IF A DEVICE IS SPECIFIED
441          BEQ 15
442          MOV @#SWR,@#DEV I GET SELECTED DEVICE
443          CLR @#EABITS I CLEAR EABITS FOR DEVICE
444          PC,IODEV I GO RELOCATE VIA SELECTED DEVICE
445          BVC 25 I '!V' =0/1 INDICATES NO ERROR/ERROR
446          MOV (R0+),(R2)+ I RELOCATE PROGRAM CODE
447          CMP R0,R3 I CHECK IF DONE
448          BNE 15
449          CMP =(R0),=(R2) I CHECK THAT CODE WAS RELOCATED
450          REQ 35 I PROPERLY
451          JSR PC,SAVVAL I GO SAVE PERTINENT DATA FOR TYPEOUT
452          HLT 104400 I ERROR! CODE NOT RELOCATED PROPERLY
453          CMP R0,R5 I CHECK IF FINISHED CHECKING
454          BNE 25
455          SUB #10,@#DEVID I BRANCH IF ERROR DETECTED ON RELOCATION
456          BEQ 115
457          INCB @#DEV I STEP TO NEXT DEVICE
458          CLR @#DEV10 I SET DEVICE IND TO CP
459          MOV R2,PC I GO EXECUTE RELOCATED CODE
460          MOV (PC),PC I RETURN TO NEXT SECTION OF CODE
461          RETPC: B I CONTAINS PC OF NEXT SECTION OF CODE
462
463          I WAIT LOOP FOR COMPLETION OF DEVICE TRANSFERS
464          WAIT1: MOV @#PC+,R4 I GET CONTENTS OF DEVICE'S BUS
465          BUSADR: WORD 0 I ADDRESS REGISTER
466          TSTB @#MNON I BRANCH IF MEM MGMT IS NOT ON
467          BEQ 15
468          BIC #160000,R4 I CONVERT ADDRESS TO VIRTUAL ADRS
469          BIS @#40000,R4
  
```

```

470          CMP =(R4),(R4)
471          BEQ 25
472          HLT 104400
473          ADD @#,(R4)
474          BR WAIT1
  
```

```
475 ;SBTTL IODEV ROUTINE
476 ;ROUTINE TO RELOCATE PROGRAM CODE VIA DEVICE SELECTED IN SWITCHES<2-8>
477 ;(IF SW03=1) OR VIA ALL DEVICES IF SW05=1,
478 ;THIS ROUTINE WRITES THE DATA TO RELOCATED ONTO THE SELECTED
479 ;DEVICE AND AFTER COMPLETION READS THE DATA BACK INTO MEMORY WHERE
480 ;THE RELOCATED DATA IS TO GO, AFTER THE READ THE ROUTINE RETURNS TO
481 ;THE CALLER, THE CALLER COMPARES THE DATA READ BACK,
482 ;DEVICES ARE:
483 ; 0-CP ;TAKES ERROR EXIT
484 ; 1-RK
485 ; 2-RF
486 ; 3-RP
487 ; 4-RC
488 ; 5-DO NOT USE
489 ; 6-RS04
490 ; 7-CP ;RESERVED FOR FUTURE USE
491 ;INPUT PARAMETERS:
492 ; R0 ;BUS ADDRESS FOR WRITE
493 ; R1 ;I/O UNIT CARE
494 ; R2 ;BUS ADDRESS FOR READ
495 ; R3 ;I/O UNIT CARE
496 ; R4 ;BYTE COUNT
497 ; R5 ;I/O UNIT CARE
498 ; EABITS ;I/O UNIT CARE
499 ; DEV ;DEVICE IDENTIFIER
500 ;OUTPUT
501 ; R0 ;UPDATED BY BYTE COUNT (IF NO ERROR)
502 ; R1 ;UNCHANGED
503 ; R2 ;UPDATED BY BYTE COUNT (IF NO ERROR)
504 ; R3 ;UNCHANGED
505 ; R4 ;CLOBBERED
506 ; R5 ;UNCHANGED
507 ; EABITS ;UNCHANGED
508 ; 'V' BIT ;CLEAR/SET=NO ERROR/ERROR
509
510 ;IODEV: JSR PC,CLRTBIT ;CLEAR 'V' BIT & SAVE PSW
511 ; MOV R5,(SP) ;SAVE R5 ON THE STACK
512 ; #RPT4,#PSW ;SET PRIORITY LEVEL 4
513 ; BIC #370,#DEV ;LIMIT DEVICE SELECT CODE
514 ; BICB #DEV,R5 ;GET SELECTED DEVICE
515 ; MOV R5 ;IFORM INDEX POINTER
516 ; ASL DEVB,(R5),R5 ;GET SELECTED DEVICE TABLE
517 ; MOV DEVB,(R5),R5 ;IFORM INDEX POINTER
518 ; BNE 2$ ;BRANCH IF I/O DEVICE SELECTED
519
520 ;99$: JSR PC,#RESTPS ;ERROR EXIT
521 ; MOV (SP)+,R5 ;RESTORE ORIGINAL PSW
522 ; SEV ;RESTORE R5
523 ; RTS PC ;SET 'V' BIT TO INDICATE FAILURE
524 ; RETURN
525
526 ;CHECK IF USER SELECTED DEVICE IS AVAILABLE
527 ;2$: MOV #ERRVEC+2,#ERRVEC ;SET TIME OUT TRAP VECTOR
528 ; SEC ;SET 'C' IN PSW
```

```
529 ;001500 005775 000012 ;TST #12(R5) ;REFERENCE A DEVICE REG
530 ;001504 012737 005274 000004 ;MOV #ERRRT,#ERRVEC ;RESTORE ERROR TRAP VECTOR
531 ;001512 103020 ;BCC 20$ ;NOTE! MOV DOES NOT AFFECT 'C'
532 ;001514 032737 000040 177570 ;BIT #40,#SWR ;EVER ONWARD IF IT'S THERE
533 ;001522 001403 ;BEO 14$ ;DO NOT REPORT ERROR IF ALL
534 ;001524 105237 000757 ;INCB #DEV ;STEP TO NEXT DEVICE
535 ;001530 000741 ;BR 1$
536 ;001532 113705 000757 14$: MOV R5 ;GET DEVICE ID
537 ;001536 006305 ;ASL R5 ;IFORM INDEX VALUE
538 ;001540 016567 004156 030602 ;MOV DEVB,(R5),DEVNAM ;GET HIS NAME
539 ;001546 000004 032346 ;TYPE,NODEV
540 ;001552 000741 ;BR 99$ ;TAKE ERROR EXIT
541
542 ;001554 112737 000003 000760 20$: MOV R4,(PC)+ ;SET ERROR RETRY COUNT
543 ;001562 010427 ;MOV #WORD ;SAVE BYTE COUNT
544 ;001564 000000 ;WORD 0
545 ;001566 010446 10$: MOV R4,-(SP) ;IFORM TWO'S COMPLEMENT
546 ;001570 006216 ;ASR (SP) ;WORD COUNT
547 ;001572 005416 ;NEG (SP)
548 ;001574 012667 ;MOV (SP)+,9$ ;AND SAVE IN 9$ BELOW
549 ;001600 113737 000757 000772 ;MOV #DEV,#DEVID ;SET DEVICE IDENT
550 ;001606 013727 000762 ;MOV #EABITS,(PC)+ ;SAVE EABITS IN 11$ BELOW
551 ;001612 000000 11$: ;WORD 0
552 ;001614 123727 000757 000004 ;CMPB #DEV,#4 ;BRANCH IF DEVICE IS NOT
553 ;001622 003415 ;BLE 12$ ;A MASS BUS DEVICE
554 ;001624 006367 177762 ;ASL 11$ ;SHIFT EA BITS TO
555 ;001630 006367 177756 ;ASL 11$ ;POSITION 8-9
556 ;001634 006367 177752 ;ASL 11$ ;FROM 4-5
557 ;001640 006367 177746 ;ASL 11$
558 ;001644 012735 000021 ;MOV #21,(R5)+ ;DO A READ IN (TO SET VOL VAL ID)
559 ;001650 012735 010000 ;MOV #10000,(R5)+ ;SET PDP11 FORMAT (IN RPOF REG)
560 ;001654 012535 ;MOV (R5)+,(R5)+ ;LOAD DEVICES UNIT #
561 ;001656 012546 ;MOV (R5)+,-(SP) ;GET DEVICE'S VECTOR ADDRESS
562 ;001660 012776 001732 000000 ;MOV #4,(SP) ;LOAD VECTOR
563 ;001666 062716 000002 ;ADD #2,(SP)
564 ;001672 012736 000240 ;#PRTY5,(SP)+ ;AND PSW ON INTERRUPT
565 ;001676 004767 000222 ;JSR PC,DSKADR ;GET RANDOM DISK ADDRESS
566 ;001702 016735 000306 3$: MOV TRKADR,(R5)+ ;SET 'CYLINDER' ADDRESS
567 ;001706 016775 000310 ;MOV TRKSEC,(R5)+ ;SET 'TRACK/SECTOR' ADDRESS
568 ;001712 011537 001364 ;MOV R0,#BUSADR ;SAVE ADDRESS OF BUS ADDRESS REG
569 ;001716 010035 ;MOV R0,(R5)+ ;SET BUS ADDRESS
570 ;001720 016735 000176 ;MOV 9$,(R5)+ ;SET WORD COUNT
571 ;001724 016535 000002 ;MOV 2(R5),(R5)+ ;AND SET COMMAND
572 ;001730 000614 30$: RR ;GO WAIT FOR WRITE TO FINISH
573
574 ;001732 012710 001740 4$: MOV #41,(SP) ;ADJUST RETURN PC TO 41$ BELOW
575 ;001736 000002 ;RTI
576 ;001740 015544 41$: MOV #-R5,R4 ;GET AND CHECK ERROR BIT
577 ;001742 100011 ;RPL 5$ ;BRANCH IF NO ERROR
578 ;001744 104400 ;HLT ;REPORT ERROR
579 ;001746 016535 000006 ;MOV 6(R5),(R5)+ ;RESET DEVICE'S CONTROLLER
580 ;001752 162705 000012 ;SUB #12,R5 ;RESET TABLE POINTER
581 ;001756 105337 000760 ;DECB #IORETRY ;RETRY WRITE COMMAND
```

```

DC0KCD 11/40-11/45 CPU EXERCISER MACY11 27(655) 4-SEP-74 11:53 PAGE 15
DC0KCD 10DEV ROUTINE
583 001762 001347          BNE 35
584 001764 000634      485:  RR      995          ;TAKE ERROR EXIT
585                                     ;AFTER THREE RETRYs
586
587 001766 112737 000003 000760 55:  MOVB  #3, #IORETRY          ;RESET ERROR RETRY COUNT
588 001774 162705 000012 65:  SUB  #12, R5          ;RESET TABLE POINTER
589 002000 012735 002046      MOV  #75, *(R5)+          ;RESET DEVICE'S INT VECTOR
590 002004 016735 000224      MOV  CYLADR, *(R5)+          ;GET 'CYLINDER' ADDRESS
591 002010 016735 000206      MOV  TRKSEC, *(R5)+          ;GET 'TRACK/SECTOR' ADDRESS
592 002014 011537 001364      MOV  (R5), #BUSADR          ;SAVE ADDRESS OF BUS ADDRESS REG
593 002020 010235          MOV  R2, *(R5)+          ;SET BUS ADDRESS
594 002022 016735 000074      MOV  #95, *(R5)+          ;SET WORD COUNT
595 002026 016746 177560      MOV  #115, *(R5)+          ;SET EA BITS
596 002032 056916 000004      BIS  4(R5), *(R5)          ;SET IN READ COMMAND
597 002036 012675 000000      MOV  (SP), *(R5)          ;LOAD COMMAND
598 002042 000240      NOP
599 002044 000731      RR      305          ;GO TO WAITIO VIA 305
600
601 002046 012716 002054      75:  MOV  #715, (SP)          ;ADJUST RETURN PC TO 715 BELOW
602 002052 000002          RTI
603 002054 013504      715:  MOV  *(R5)+, R4          ;GET & CHECK ERROR BIT IN COMMAND REG
604 002056 100007          BPL  85
605 002060 104400      HLT
606 002062 016555 000004      MOV  4(R5), *(R5)          ;RESET DEVICE'S CONTROLLER
607 002066 105337 000760      DECB #IORETRY          ;RETRY READ COMMAND
608 002072 001340      BNE  65
609 002074 000733      RR      485          ;3 TIMES AND IF STILL FAILS
610 002076 012605      85:  MOV  (SP)+, R5          ;TAKE ERROR EXIT
611 002100 066700 177460      ADD  #105, R0          ;RESTORE R5
612 002104 066702 177454      ADD  #105, R2          ;ADD BYTE COUNT TO WRITE AND
613 002110 004767 000442      JSR  PC, RESTPS          ;READ ADDRESSES (FOR CHECKING)
614 002114 000242          CLV          ;GO RESTORE 'I' IN PSW
615 002116 000167 177344          JMP  1005          ;CLEAR ERROR INDICATOR
616                                     ;EXIT
617 002122 000000      95:  .WORD 0          ;CONTAINS TWO'S COMP WORD COUNT
618
619 ;SUBROUTINE TO GENERATE RANDOM DISK SURFACE ADDRESSES
620 002124 032737 000020 177570 DSKADR: BIT  #20, #SWR          ;BRANCH IF USER DOES NOT WANT
621 002132 001426          BCC  25          ;RANDOM DISK ADDRESSES
622 002134 010046          MOV  R0, *(R5)          ;SAVE R0 ON THE STACK
623 002136 013700 000772      MOV  #0, DEVIO, R0          ;GET I/O DEVICE ID
624 002142 006300          ASL  R0          ;FORM INDEX INTO
625 002144 006300          ASL  R0          ;ADRTAB BELOW
626 002146 060146          ADD  R1, -(SP)          ;FORM RANDOM #
627 002150 005516          ADC  (SP)
628 002152 011667 000036      MOV  (SP)+, CYLADR          ;MOVE TO 'CYLINDER' ADDRESS
629 002156 046067 002226 000030      BIC  ADRTAB(R0), CYLADR          ;LIMIT 'CYLINDER' ADDRESS
630 002164 060116          ADD  R1, *(R5)
631 002166 005516          ADC  (SP)
632 002170 012667 000026      MOV  (SP)+, TRKSEC          ;MOVE TO 'TRACK/SECTOR' ADDRESS
633 002174 005720          YST  (R0)+
634 002176 046067 002226 000016      BIC  ADRTAB(R0), TRKSEC          ;LIMIT 'TRACK/SEC' ADRS
635 002204 012600      MOV  (SP)+, R0          ;RESTORE R0
636 002206 000207      RTS  PC          ;RETURN

```

```

DC0KCD 11/40-11/45 CPU EXERCISER MACY11 27(655) 4-SEP-74 11:53 PAGE 16
DC0KCD 10DEV ROUTINE
637 002212 012727 000000      25:  MOV  #0, (PC)+          ;SET CYLINDER ADDRESS = 0
638 002214 000000      CYLADR: .WORD 0
639 002216 012727 000000      MOV  #0, *(PC)+          ;SET TRACK & SECTOR = 0
640 002222 000000      TRKSEC: .WORD 0
641 002224 000207      RTS  PC
642
643 ;TABLE OF DEVICE 'CYLINDER' AND 'TRACK/SECTOR' ADDRESS LIMITERS
644 002226 000000      ADRTAB: .WORD 0          ;NOT USED
645 002230 000000          .WORD 0          ;NOT USED
646 002232 163350          .WORD 163350          ;RKDA LIMITER
647 002234 163350          .WORD 163350          ;RKDA LIMITER
648 002236 177774          .WORD 177774          ;RFDAE LIMITER
649 002240 020000          .WORD 020000          ;RFDAE LIMITER
650 002242 177152          .WORD 177152          ;RPCA LIMITER
651 002244 170370          .WORD 170370          ;RPDA LIMITER
652 002246 176400          .WORD 176400          ;RCDA LIMITER
653 002250 176400          .WORD 176400          ;RCDA LIMITER
654 002252 177345          .WORD 177345          ;RP4CA LIMITER
655 002254 170370          .WORD 170370          ;RP4OST LIMITER
656 002256 170400          .WORD 170400          ;RPDA LIMITER
657 002260 170400          .WORD 170400          ;RPDA LIMITER
658 002262 177777          .WORD =1          ;NOT USED
659 002264 177777          .WORD =-1          ;NOT USED
660
661 ;SBTTL DEVICE TABLES
662 002266 000000      DEVTBL: .WORD 0
663 002270 002306          .WORD RKTBL
664 002272 002330          .WORD RFTBL
665 002274 002352          .WORD RPTBL
666 002276 002374          .WORD RCTBL
667 002300 000000          .WORD 0          ;RESERVED FOR RP04
668 002302 002450          .WORD RSTBL
669 002304 000000          .WORD 0
670
671 002306 000220      RKTBL: .WORD RKVEC
672 002310 177412          .WORD RKDA
673 002312 177412          .WORD RKDA
674 002314 177410          .WORD RKBA
675 002316 177406          .WORD RKNC
676 002320 177404          .WORD RKCS
677 002322 000503          .WORD 503          ;WRITE COMMAND
678 002324 000505          .WORD 505          ;READ COMMAND
679 002326 000001          .WORD 1          ;CONTROL RESET
680
681 002330 000204      RFTBL: .WORD RFVEC
682 002332 177470          .WORD RFDAE
683 002334 177466          .WORD RFDAE
684 002336 177464          .WORD RFDAE
685 002340 177462          .WORD RFDAE
686 002342 177460          .WORD RFDAE
687 002344 000103          .WORD 103          ;WRITE COMMAND
688 002346 000105          .WORD 105          ;READ COMMAND
689 002350 000001          .WORD 1          ;CONTROL RESET
690
691

```


DC0KCD TYPE SUBROUTINE

```

762
763
764 002564 010046
765 002566 017600 00002
766 002572 062766 00002 000002
767 002600 032737 00400 000764
768 002606 001403
769
770 002610 112046 15:
771 002612 001003 RNE
772 002614 005726 TST
773 002616 012600 65:
774 002620 000002 MOV
775 RTI
776 002622 004767 000026 25:
777 002626 122726 000012 35:
778 002632 001366 CMPE
779 BNE
780 002634 016746 76142 MOV
781
782 002640 105366 00001 45:
783 002644 002770 DECB
784 002646 004767 000002 RLT
785 002652 000772 JSR
786 BR
787
788 002654 105737 177564 55:
789 002660 100375 BPL
790 002662 116637 000002 177566 MOV
791 002670 000207 RTS
792
793 NULL=0
794
795
796
797
798 002672 004767 002204 CNVDAT:
799 002676 012704 003102 JSR
800 002702 010201 MOV
801 002704 005003 MOV
802 002706 012700 000006 CLR
803 002712 000167 000100 MOV
804 JMP
805
806
807
808
809
810
811
812
813
814
815

```

SRCTL TYPE SUBROUTINE

```

ROUTINE TO TYPE ASCII MESSAGE; MESSAGE MUST TERMINATE WITH A 0 BYTE,
TYPE: MOV R0,*(SP) ;SAVE R0 ON THE STACK
MOV #2(SP),R0 ;GET MESSAGE ADDRESS
ADD #2,2(SP) ;ADJUST RETURN PC
BIT #TTOPT,0#OPT,CP ;BRANCH IF NO CONSOLE TTY AVAIL
BCD 65

15: MOVB (R0)+,*(SP) ;PUSH CHAR ON THE STACK
RNE 25 ;BRANCH IF NOT TERMINATOR
TST (SP)+ ;POP TERMINATOR OFF THE STACK
65: MOV (SP)+,R0 ;RESTORE R0
RTI ;RETURN

25: JSR PC,55 ;TYPE CHARACTER
35: CMPE #12,(SP)+ ;CHECK IF CHAR WAS A LINE FEED
BNE 15 ;BRANCH IF NOT LINE FEED

MOV 5FILLS,*(SP) ;GET # OF FILLERS REQUIRED AFTER
LINE FEED AND FILLER CHARACTER
45: DECB 1(SP) ;DECREMENT FILLERS COUNT
RLT 35 ;BRANCH IF NO MORE FILLERS NEEDED
JSR PC,55 ;TYPE FILLER CHARACTER
BR 45

55: TSTB #TPS ;WAIT FOR OUTPUT DEVICE
BPL 4 ;TO BECOME READY
MOV 2(SP),0#TPB ;TYPE CHARACTER
RTS PC

NULL=0

;SUBROUTINE TO CONVERT 16 BIT DATA TO ASCII STRING, THE ASCII STRING
;STARTS AT DIGITS AND IS 8 BYTES LONG, 6 ASCII DIGITS + 'SPACE' + '0',
CNVDAT:
JSR PC,SSAVR ;GO SAVE REGISTERS ON THE STACK
MOV #DIGBUF+0,,R4 ;SET ADDRESS OF DIGIT BUFFER
MOV R2,R1 ;GET DATA
CLR R3 ;
MOV #6,R0 ;SET DIGIT COUNT
JMP CNVDIG ;GO TO DIGIT CONVERSION ROUTINE

;SUBROUTINE TO CONVERT A VIRTUAL ADDRESS TO AN ASCII STRING PHYSICAL
;ADDRESS, THE CONVERTED ASCII STRING IS AT 'DIGBUF' AND IS 10 BYTES LONG
;(8 DIGITS + 1 SPACE + 0 BYTE)
CNVADR:
CALLI MOV ADDRESS,R1 ;GET ADDRESS
JSR PC,CNVADR ;
;NOTE: SUBROUTINE SUBTRACTS 2 FROM ADDRESS BEFORE CONVERSION
;FOR EXAMPLE TO TYPE ERROR PC
MOV PC,R1 ;IT IS THE PC OF THE MOV
JSR PC,CNVADR ;THAT GETS TYPED
TYPE ;
DIGBUF ;

```

DC0KCD TYPE SUBROUTINE

```

816
817 002716
818 002716 004767 002202 CNVADR:
819 002722 012704 003102 JSR
820 002726 162701 000002 MOV
821 002732 010105 SUB
822 002734 005003 MOV
823 002736 105737 000770 CLR
824 002742 001423 TSTB
825 002744 042701 017777 #MMON
826 002750 006301 BEQ
827 002752 006101 BIC
828 002754 006101 R1
829 002756 006101 R1
830 002760 006301 R1
831 002762 062701 172340 ADD
832 002766 011101 #KIRAR0,R1
833 002770 012700 000006 (R1),R1
834 002774 006301 25: MOV
835 002776 006103 R1
836 003000 077003 ROL
837 003002 042705 160000 SOB
838 003006 060501 BIC
839 003010 005503 ADD
840 003012 012700 000010 ADC
841
842 003016 012705 000003 MOV
843 003022 005002 CLR
844 003024 006203 55: ASR
845 003026 006001 R1
846 003030 106002 ROR
847 003032 005305 DEC
848 003034 001373 RNE
849 003036 012705 000005 MOV
850 003042 000241 65: CLC
851 003044 106002 ROR
852 003046 005305 R5
853 003050 001374 BNE
854
855 003052 062702 000260 ADD
856 003056 110244 MOV
857 003060 005300 DEC
858 003062 001355 BNE
859 003064 004767 000054 JSR
860 003070 000207 RTS
861
862
863 003072 000 000 ;DIGIT BUFFER
864 003074 000006 DIGBUF, BYTE
865 003102 040 ;BLKB
866 003103 000 ;BYTE
867 ;'0' TERMINATOR
868
869

```

```

;AND BITS PER DIGIT COUNT
;R2 WILL CONTAIN DIGIT
;R3<00> TO '0'
;'0' TO R1<15> & R3<00> TO '0'
;'0' TO R2<07>
;DECREMENT SHIFT COUNT
;SET SHIFT COUNT
;SHIFT DIGIT FROM <07-03>
;TO <02-00>

;CONVERT DIGIT TO ASCII
;MOVE DIGIT INTO DIGIT BUFFER
;DECREMENT DIGIT COUNT
;CONVERT NEXT DIGIT
;RESTORE REGISTERS FROM STACK

;LOAD R2 WITH THE DATA

```

```
872 ; JSR PC,TYPDAT
871
872 TYPDAT: JSR PC,CNVDAT ;CONVERT DATA TO ASCII STRING
873 TYPE,DIGITS
874 RTS PC
875
876 ;SUBROUTINE TO CONVERT A VIRTUAL ADDRESS TO A PHYSICAL ADDRESS AND TYPE IT,
877 ;CALL: MOV #ADDRESS,R1 ;LOAD R1 WITH THE ADDRESS
878 ; JSR PC,TYPADR
879
880 TYPADR: JSR PC,CNVADR ;CONVERT ADDRESS TO ASCII STRING
881 TYPE,DIGBUF ;TYPE ADDRESS
882 RTS PC
883
884 ;KEYBOARD INTERRUPT SERVICE ROUTINE
885
886 CNTRLC=3
887
888 TKISR: NOP
889 MOV #TKB,(SP) ;GET CHARACTER
890 BIC #177600,(SP) ;STRIP UNUSED BITS
891 CMP #CNTRLC,(SP) ;BRANCH IF NOT CONTROL C (*C)
892 BNE 1$
893 TYPE,CRLF ;ECHO <CR><LF>
894 TST (SP)+ ;POP CHARACTER OFF THE STACK
895 HALT
896 RTI ;RETURN
897
898 1$: CMPB #15,(SP) ;BRANCH IF NOT <CR>
899 BNE 2$
900 TYPE,CRLF ;ECHO <CR><LF>
901 TST (SP)+ ;POP CHARACTER OFF STACK
902 RTI ;RETURN
903
904 2$: MOVB (SP)+,ECHO ;ECHO CHARACTER
905 TYPE,ECHO ;RETURN
906 RTI
```

```
907 ;@TTL ERROR SERVICE ROUTINE
908 ;ERROR SERVICE CALLED BY TRAP (HLT) INSTRUCTION
909 .HLT TST #SWR ;HALT ON ERROR?
910 BPL ;+4
911 HALT ;ERROR PC IS TOP WORD ON STACK
912 BIT #20000,#SWR ;TYPE OUT DESIRED?
913 BNE 1$ ;BRANCH IF NO TYPEOUT
914 JSR PC,$$AVR ;GO SAVE REGISTERS ON THE STACK
915 MOV #CNT,R2 ;GET PASS COUNT
916 JSR PC,CNVDAT ;LOAD ASCII VALUES
917 MOV DIGITS*2,PASSES
918 MOV DIGITS*4,PASSES*2
919 TYPE,PASCNT
920 MOV 16(SP),R2 ;GET PC OF ERROR CALL
921 CMPB #(R2),=(R2) ;DECREMENT PC TO HLT
922 TYPE,VIRPC
923 JSR PC,TYPDAT ;TYPE DATA
924 MOV DEVID,R2 ;GET DEVICE IDENTIFICATION
925 BEQ 13$ ;AND BRANCH IF DEVICE WAS CP
926 ASL R2
927 MOV DEVICE(R2),DEVERR
928 TYPE,DEVERR
929 JSR PC,RNTREGS
930 BR 19$
931 13$: TYPE,STATUS
932 MOV 20(SP),R2 ;GET STATUS AT TIME OF ERROR
933 JSR PC,TYPDAT ;TYPE STATUS
934 CMPB #10,#OPT,CP
935 BNE 12$
936 TYPE,CPERR
937 MOV #CPUERR,R2
938 JSR PC,TYPDAT
939 TYPE,ERREG
940 MOV #ERRREG,R2
941 JSR PC,TYPDAT
942 MOV 16(SP),R2 ;GET PC OF ERROR
943 CMPB #(R2),=(R2)
944 TST #MMON ;CHECK IF MEM MGMT IS ENABLED
945 BNE 10$ ;BRANCH IF ENABLED
946 TST #FACTOR
947 BEQ 19$
948
949
950 TYPE,RELPC
951 SUB #FACTOR,R2 ;FORM PC OF ORIGINAL CODE
952 JSR PC,TYPDAT ;TYPE DATA
953 BR 19$ ;GO TO 19$
954 10$: TYPE,PHYSPC
955 MOV 16(SP),R1 ;GET ERROR PC
956 JSR PC,TYPADR ;TYPE ADDRESS
957
958 19$: JSR PC,$$RSTR ;RESTORE REGISTERS FROM STACK
959 BIT #2000,#SWR ;RING BELL ON ERROR
960 BEQ 2$
```

```
961 003500 000004 01640 TYPE,BELL
962 003504 005737 177570 25: TST #ASHR HALT AFTER TYPEOUT
963 003510 100001 BPL ,#4
964 003512 000000 HALT
965 003514 005737 005322 TST #WERFLAG
966 003520 001407 BEQ #5
967 003522 005737 005322 CLR #WERFLAG
968 003526 005000 R0
969 003530 005300 R0 ALLOW TIME FOR TTY TO TYPE CHAR
970 003532 001376 RNE #5
971 003534 00137 005422 JMP #START
972 003540 105737 000761 45: TSTB #WERFLG JBRANCH IF NO PARITY ERROR
973 003544 001402 BEQ #5
974 003546 000137 004402 JMP #PERET JRETURN TO PARITY ERROR SERVICE
975 003552 000002 55: RTI
976
977
978 003554 030440 IDIGIT TABLE
979 003556 031442 DIGTAB: #01
980 003560 032444 #23
981 003562 033446 #49
982 003564 005015 040520 051523 PASCNT: #ASCII <15><12>#PASS #1
983 003572 021440 #47
984
985
986 003574 030060 030060 000 PASSES: #ASCII '0000'
987 003601 040 050126 036503 VIRPC: #ASCII 'VPO'
988 003604 000
989 003607 120 053523 000075 STATUS: #ASCII 'PSNR'
990 003614 050103 036025 000 OPERR: #ASCII 'CPUR'
991 003621 105 051122 000075 ERRR: #ASCII 'ERRR'
992 003626 050122 036003 000 RELPC: #ASCII 'RPRR'
993 003633 120 041520 000075 PHYSPC: #ASCII 'PRPR'
994 003640 000007 BELL: #ASCII '<7>'
995
996
997 003642 005015 SUCCESS: #ASCII <15><12>
998 003644 052040 242510 350440 #ASCII / THE QUICK BROWN FOX JUMPS OVER THE LAZY DOGS BACK #123456789 PASS# /
999 003652 044525 045503 041040
1000 003660 047522 047127 043040
1001 003666 054117 045040 046525
1002 003674 051520 047440 042526
1003 003702 020122 044124 020105
1004 003710 040514 054532 042040
1005 003716 043517 020123 040502
1006 003724 045503 030040 031061
1007 003732 032063 033065 034067
1008 003740 020071 040520 051523
1009 003746 020043
1010 003750 030060 030060 000 PASSNO: #ASCII '0000'
1011 #EVEN
1012 #ROUTINE TO TYPE CONTENTS OF DEVICE REGISTER ON AN ERROR
1013 #INPUT:
1014 # R2 #INDEX VALUE TO APPROPRIATE DEV
```

```
1015 003756 016200 004200 PNTREGS: MOV REGS(2),R0 JGET # OF REGS TO TYPE
1016 003762 016203 004222 MOV REGADR(2),R3 JGET FIRST ADDRESS OF DATA TABLE
1017 003766 022703 020730 CMP #HEHTBL,R3 JBRANCH IF MEMORY ERROR
1018 003772 001406 REG 25
1019 003774 012302 15: MOV (R3),R2
1020 003776 004767 177102 JSR PC,VYPDAT JTYPE DATA
1021 004002 005300 R0
1022 004004 001373 R0
1023 004006 000207 RTS PC
1024
1025 004010 000004 004066 25: TYPE,GOADR
1026 004014 012301 MOV (R3),R1 JGET 'FROM' ADDRESS
1027 004016 005721 TST (R1) JADD 2
1028 004020 004767 177072 JSR PC,VYPADR JTYPE ADDRESS
1029 004024 000004 TYPE,A,DATA
1030 004030 012302 MOV (R3),R2 JGET 'FROM' DATA
1031 004032 004767 177046 JSR PC,VYPDAT JTYPE DATA
1032 004036 000004 TYPE,B,DATA
1033 004042 012301 MOV (R3),R1 JGET 'TO' ADDRESS
1034 004044 005721 TST (R1) JADD 2
1035 004046 004767 177044 JSR PC,VYPADR JTYPE ADDRESS
1036 004052 000004 TYPE,A,DATA
1037 004056 012302 MOV (R3),R2 JGET 'TO' DATA
1038 004060 004767 177020 JSR PC,VYPDAT JTYPE DATA
1039 004064 000207 RTS PC
1040
1041 004066 051105 047522 020122 GOADR: #ASCII 'ERROR ON PROGRAM RELOCATION'<15><12>
1042 004074 047117 050040 047522
1043 004102 051107 046501 051040
1044 004110 046105 041517 052101
1045 004116 047511 006916 012
1046 004123 107 047917 020104 #ASCII 'GOOD ADRS#'
1047 004130 042101 051522 000075 A,DATA: #ASCII 'DATA#'
1048 004136 040504 040524 000075 B,DATA: #ASCII 'BAD ADRS#'
1049 004144 040502 020104 042101
1050 004152 051522 020075 #EVEN
1051
1052
1053 004156 030060 DEVICE: #ASCII '#0'
1054 004160 045522 #ASCII '#RK'
1055 004162 043122 #ASCII '#RF'
1056 004164 050122 #ASCII '#RP'
1057 004166 041522 #ASCII '#RC'
1058 004170 050122 #ASCII '#RP'
1059 004172 051522 #ASCII '#RS'
1060 004174 054130 #ASCII '#XX' JRESERVED FOR FUTURE USE
1061 004176 046515 #ASCII '#MM' MEMORY
1062
1063 #THE BELOW TABLE CONTAINS THE # OF DEVICE REGISTERS TO TYPE ON A
1064 #DEVICE ERROR
1065 # REGS: #WORD 1 #NOT USED (FOR CP)
1066 # #WORD 6 #TYPE 6 RK REGISTERS
1067 # #WORD 6 #TYPE 6 RF REGISTERS
1068 # #WORD 8, #TYPE 8, RP REGISTERS
1069 # #WORD 6 #TYPE 6 RC REGISTERS
```

```

1069 004212 000011          WORD 9,          ;TYPE 9 RP04 REGISTERS
1070 004214 000011          WORD 9,
1071 004216 000001          WORD 1,
1072 004220 000004          WORD 4,
1073
1074 004222 000000          REGADR: WORD 0
1075 004224 177400          WORD RKDS
1076 004226 177460          WORD RFDGS
1077 004230 176710          WORD RPDS
1078 004232 177440          WORD RCLA
1079 004234 176700          WORD RP4CS1
1080 004236 172040          WORD RSCS1
1081 004240 000000          WORD 0
1082 004242 000730          WORD MEMTBL
1083
1084
1085
;ROUTINE TO GET TYPED OCTAL ADDRESS AND CONVERT TO OCTAL, CALL:
1086 ; JSR R5,RECO
1087 ; WORD 0 ;CONVERTED DATA IS PLACED HERE
1088 004244 010046          RECO: MOV R0,=(SP) ;SAVE R0 ON THE STACK
1089 004246 005015          CLR (R5) ;CLEAR OLD DATA
1090 004250 105737 177560 1$; TSTB @R5TKS ;WAIT FOR USER TO INPUT CHARACTER
1091 004254 100375          BPL ;
1092 004256 113700 177562          MOV @TKB,R0 ;GET CHARACTER
1093 004262 042700 000200          BIC #200,R0 ;STRIP MSB
1094 004266 122700 000177          CMPB #177,R0 ;CHECK IF RUBOUT
1095 004272 001007          BNE ;BRANCH IF NOT RUBOUT
1096 004274 000004 000755          TYPE,SLASH ;ECHO SLASH
1097 004300 000241          CLC ;CLEAR CARRY
1098 004302 000015          ROR (R5) ;SHIFT LAST TYPED CHARACTER
1099 004304 000215          ASR (R5) ;OUT OF DATA WORD
1100 004306 000215          ASR (R5)
1101 004310 000757          BR 1$ ;GO WAIT FOR NEXT CHARACTER
1102
1103 004312 122700 000015 2$; CMPB #15,R0 ;CHECK IF CARRIAGE RETURN
1104 004316 001004          BNE ;BRANCH IF NOT CARRIAGE RETURN
1105 004320 000004 000752          TYPE,CRLF
1106 004324 005725          TST (R5)+ ;STEP RETURN ADDRESS
1107 004326 000205          RTS R5 ;RETURN
1108
1109 004330 110067 174404 3$; MCVB R0,ECHO
1110 004334 000004 000740          TYPE,ECHO
1111 004340 042700 177770          BIC #177770,R0 ;STRIP NON-ESSENTIAL BITS
1112 004344 000315          ASL (R5) ;SHIFT LAST CHARACTER 3 PLACES
1113 004346 000315          ASL (R5) ;LEFT
1114 004350 000315          ASL (R5)
1115 004352 050015          BIS R0,(R5) ;AND INSERT NEW CHARACTER
1116 004354 000735          BR 1$ ;WAIT FOR NEXT CHARACTER
1117

```

```

1118          ;SBTTL PARITY ERROR SERVICE
1119 ;PARITY ERROR SERVICE ROUTINE
1120 ;PARSRVITST @SWR ;CHECK IF HALT ON ERROR
1121 004356 005737 177570          BPL 1$ ;BRANCH IF NOT HALT ON ERROR
1122 004364 000000          HALT
1123 004366 000004 000707 1$; TYPE,PARERRR
1124 004372 110637 000761          MOVB SP,@PEFLG ;SET PARITY ERROR INDICATOR
1125 004376 000137 003212          JMP @,HALT ;GO TO ERROR SERVICE
1126 004402 105037 000761          PERET: CLRB @PEFLG ;CLEAR PARITY ERROR FLAG
1127 004406 005001          CLR R1
1128 004410 005737 000764          TST @PORT,Cp ;CHECK IF MEM MGMT IS AVAIL
1129 004414 100032          BPL 1$ ;BRANCH IF NOT AVAILABLE
1130 004416 012702 077406          MOV #77406,R2 ;SET UP MEM MGMT
1131 004422 005037 172340          CLR @KIPAR0
1132 004426 010237 172300          MOV R2,@KIPDR0
1133 004432 012737 000200 172342          MOV #200,@KIPAR1
1134 004440 010237 172302          MOV R2,@KIPDR1
1135 004444 012737 000400 172344          MOV #400,@KIPAR2
1136 004452 010237 172304          MOV R2,@KIPDR2
1137 004456 005037 172306          CLR @KIPDR3
1138 004462 012737 007600 172356          MOV #7600,@KIPAR7
1139 004470 010237 172316          MOV R2,@KIPDR7
1140 004474 012737 000001 177572          MOV #1,@SR0 ;ENABLE MEM MGMT
1141 004502 012737 004530 000114 1$; MOV #5,@ERRVEC ;SET NEW PARITY ERROR TRAP VECTOR
1142 004510 012737 004710 000004          MOV #75,@ERRVEC ;SET TIME OUT TRAP
1143 004516 012737 004722 000250          MOV #05,@MHVEC ;SET MEM MGMT ABORT VECTOR
1144
1145 004524 005721          TST (R1)+ ;SCAN MEMORY FOR PARITY ERROR
1146 004526 000776          BR -2
1147
1148 004530 000004 004744 2$; TYPE,ADRSIS
1149 004534 004767 176356          JSR PC,TYPADR ;TYPE ADDRESS
1150 004540 000005          RESET ;DISABLE PARITY ERROR DETECTION & MEM MGMT
1151 004542 005737 000764          TST @PORT,Cp ;BRANCH IF MEM MGMT NOT AVAILABLE
1152 004546 100002          BPL 3$
1153 004550 005237 177572          INC @SR0 ;RE-ENABLE MEM MGMT
1154 004554 005002          CLR R2 ;INITIALIZE DATA FOR DATA SCAN
1155 004556 014103          MOV -(R1),R3 ;GET DATA IN FAILING ADDRESS
1156 004560 010211 4$; MOV R2,@R1 ;LOAD BINARY COUNT INTO ADDRESS
1157 004562 021102          CMP (R1),R2 ;BRANCH IF DATA DOES NOT COMPARE
1158 004564 001016          BNE 5$
1159 004566 005102          COM R2 ;COMPLEMENT DATA
1160 004570 010211          MOV R2,(R1) ;LOAD COMPLEMENT DATA INTO FAILING ADDRESS
1161 004572 021102          CMP (R1),R2 ;BRANCH IF DATA DOES NOT COMPARE
1162 004574 001012          BNE 5$
1163 004576 005402          NEG R2 ;STEP DATA
1164 004600 001367          BNE 4$
1165 004602 000004 004771          TYPE,NOTFND ;TYPE PARITY ERROR NOT FOUND ON
1166 004606 000004 005046          TYPE,ISCAN ;DATA SCAN ORIG DATA =
1167 004612 010302          MOV R3,R2 ;GET ORIGINAL DATA
1168 004614 004767 176264          JSR PC,TYPDAT ;TYPE ORIGINAL DATA
1169 004620 000411          BR 6$ ;EXIT VIA 6$
1170
1171 004622 000004 005075 5$; TYPE,GODAT ;TYPE GOOD DATA =

```



```

1172 004626 004767 176252 JSR PC,TYPE DAT AND THE GOOD DATA
1173 004632 000004 005110 TYPE,BDDAT ITYPE BAD DATA =
1174 004636 011102 MOV (R1),R2 IGET BAD DATA
1175 004640 004767 176240 JSR PC,TYPE DAT ITYPE BAD DATA
1176
1177 004644 000004 000752 65I TYPE,CRLF
1178 004650 005737 177570 TST #R SWR I CHECK FOR HALT ON ERROR
1179 004654 100021 GBL #4
1180 004656 000000 HALT
1181 004660 000005 RESET I DISABLE MEM MGMT & PARITY
1182 004662 012737 004356 000114 MOV #,PARSRV,##PARVEC I RESET PARITY ERROR TRAP
1183 004670 012737 005274 000004 MOV #ERRRT,##ERRVEC I AND ERROR VECTOR
1184 004676 012737 000252 000250 MOV #HMVEC-2,##HMVEC I RESET MEM MGMT ABORT TRAP
1185 004704 000137 005600 JMP #START3 I RESTART TEST
1186
1187 004710 000004 004771 75I TYPE,NOTFND
1188 004714 000004 005031 TYPE,ASCAN BR 68
1189 004720 000751
1190
1191 MEMORY MANAGEMENT ABORT ROUTINE
1192 004722 062737 000200 172344 65I ADD #200,##KIPAR2 I ADJUST PHYSICAL ADDRESS
1193 004730 012701 020000 MOV #20000,R1 I RESET VIRTUAL ADDRESS
1194 004734 012737 000001 177572 MOV #1,##SR0 I RESET ERROR AND ENABLE
1195 004742 000002 RTI I RETURN
1196
1197 004744 005015 042515 047515 ADRSIS: ASCIZ '<15><12>'MEMORY ADDRESS IS '
1198 004752 054522 040440 042104
1199 004760 042522 051523 044440
1200 004766 020123 000
1201 004771 015 050012 051101 NOTFND: ASCIZ '<15><12>'PARITY ERROR NOT DETECTED ON '
1202 004776 052111 020131 051100
1203 005004 047522 020122 047516
1204 005012 020124 042504 042524
1205 005020 052103 042105 047440
1206 005026 020116 000
1207 005031 101 042104 042522 ASCAN: ASCIZ 'ADDRESS SCAN'
1208 005036 051523 051440 040503
1209 005044 000116
1210 005046 040504 040524 051440 DSCAN: ASCIZ 'DATA SCAN ORIG DATA = '
1211 005054 040503 020116 051117
1212 005062 043511 042040 052101
1213 005070 020101 020075 000
1214 005075 040 042107 042040 GDOAT: ASCIZ 'GD DATA = '
1215 005102 052101 034501 000040
1216 005110 041040 020104 040504 BDOAT: ASCIZ 'BD DATA = '
1217 005116 040524 020075 000
1218 005124 000 EVEN

```

```

1219 MISC SUBROUTINES
1220 ROUTINE TO SAVE REGISTERS ON THE STACK
1221 CALLED BY SAVE MACRO OR JSR PC,$SAVR
1222 005124 010546 SSAVR: MOV X0,(SP)
1223 005126 010446 MOV X4,(SP)
1224 005130 010346 MOV X3,(SP)
1225 005132 010246 MOV X2,(SP)
1226 005134 010146 MOV X1,(SP)
1227 005136 010046 MOV X0,(SP)
1228 005140 016007 000014 MOV 14(SP),PC I RETURN
1229
1230 ROUTINE TO RESTORE REGISTERS SAVED ON THE STACK
1231 CALLED BY RESTORE MACRO OR JSR PC,$RESTR
1232 005144 012666 000014 $RESTR: MOV (SP)+,14(SP) I SAVE RETURN PC
1233 005150 012600 MOV (SP)+,X0
1234 005152 012501 MOV (SP)+,X1
1235 005154 012402 MOV (SP)+,X2
1236 005156 012303 MOV (SP)+,X3
1237 005160 012204 MOV (SP)+,X4
1238 005162 012105 MOV (SP)+,X5
1239 005164 000207 RTS PC
1240
1241 SUBROUTINE TO LOAD DISPLAY REGISTER
1242 LDDISPI MOV #IGNT,(PC)+ I LOAD PASSCOUNT
1243 005172 000000 DISPLAY: WORD 0
1244 005174 012746 MOV (PC)+,(SP) I GET SECTION #
1245 005176 000000 SECT: WORD 0
1246 005200 006316 ASL (SP)
1247 005202 006316 ASL (SP)
1248 005204 006316 ASL (SP)
1249 005206 052607 B18 (SP)+,DISPLAY I LOAD SECTION #
1250 005212 113707 001011 177753 MOV #FRSTAD-1,DISPLAY+1 I LOAD BASE ADDRESS
1251 005220 105737 000770 TSTB #MMON I CHECK IF MEM MGMT IS ON
1252 005224 001403 BEQ 1$ I BRANCH IF OFF
1253 005226 013737 172344 005172 MOV ##KIPAR2,##DISPLAY I LOAD CONTENTS OF KIPAR2
1254 005234 013737 005172 177750 15I MOV ##DISPLAY,##DISPLAY I DISPLAY IN DISPLAY REGISTER
1255 005242 000207 RTS PC I RETURN
1256

```

```

1257          ;SBTTL  KT ABORT, RESERVED & ERROR TRAP SERVICE
1258          JMEMORY MANAGEMENT ABORT SERVICE ROUTINE
1259 005244 012737 005357 005326 KTABRTI  MOV  #KTAMSG,#ERTAG ;SET UP KT11 ABORT MSG
1260 005252 013716 177576          MOV  #SR2,(SP)      ; SR2 ONTO STACK
1261 005256 062716 000002          ADD  #2,(SP)
1262 005262 000410          BR   ERRPRT
1263
1264          ;RESERVED INSTRUCTION TRAP SERVICE ROUTINE
1265 005264 012737 005374 005326 RESERRI  MOV  #RESMSG,#ERTAG ;LOAD RESERVED TRAP MESSAGE
1266 005272 000412          BR   ERRPRT
1267
1268          ;TRAP TO 4 ERROR SERVICE ROUTINE
1269 005274 012737 000340 177776 ERPTI   MOV  #PRTY7,#PSW   ;SET PRIORITY LEVEL 7
1270 005302 005737 005322          TST  #ERFLAG      ;CHECK IF LAST ERROR TRAP HAS BEEN
1271 005306 001401          BEQ  ,+4          ;REPORTED
1272 005310 000000          HALT             ;ERROR! TRAPPING TO LOCATION 4
1273
1274          ;STACK CONTENTS:
1275 I (SP)          ;THIS TRAP PC
1276 I 2(SP)        ;THIS TRAP PSW
1277 I 4(SP)        ;FIRST TRAP PC
1278 I 6(SP)        ;FIRST TRAP PSW
1279 005312 012737 005340 005326          MOV  #ERMSG,#ERTAG ;SET UP TIME OUT TRAP MSG
1280 005320 005227          ERRPRTI INC  #PC+
1281 005322 000000          ERFLAGI WORD 0
1282 005324 000004          TYPE
1283 005326 000000          ERTAGI  WORD 0      ;CONTAINS ADR OF ERROR MSG
1284 005330 005037 000772          CLR  #DEVID       ;SET DEVICE ID = CP
1285 005334 000137 003212          JMP  #,BLT
1286
1287 005340 005015 051124 050101 ERMSGI  ,ASCIZ '<15><12> !TRAPPED TO 4'
1288 005346 042520 020104 047324
1289 005354 032040 000
1290 005357 015 045412 030524 KTAMSGI ,ASCIZ '<15><12> !KT11 ABORT!'
1291 005364 020061 041101 051117
1292 005372 000124
1293 005374 005015 042522 042523 RESMSGI ,ASCIZ '<15><12>'RESERVED INST TRAP'
1294 005402 053122 042105 044440
1295 005410 051516 020124 051124
1296 005416 050101 000
1297          ;EVEN
1298

```

```

1299          ;SBTTL  PROGRAM INITIALIZATION
1300 005422 000005          STARTI  RESET
1301 005424 012706 000600          MOV  #KPTR,SP      ;SET KERNEL STACK PTR
1302
1303          ;DETERMINE IF PROGRAM LOADED VIA ACT11 IN QUICK VERIFY MODE
1304 005430 105037 000771          CLR  #QV          ;SET IND NOT QV MODE
1305 005434 005737 000042          TST  #Q2          ;BRANCH IF NOT VIA ACT11
1306 005440 001405          BEQ  ,+15         ;BRANCH IF NOT QV
1307 005442 005737 032174          TST  #LOGICAL+2
1308 005446 100002          BPL  ,+15         ;BRANCH IF NOT QV
1309 005450 110637 000771          MOV  SP,#QV       ;SET ACT11 QV MODE
1310
1311 005454 012737 005476 000004 ;ROUTINE TO DETERMINE LAST MEMORY ADDRESS
1312 005462 012737 000002 15I  MOV  #25,#ERRVEC ;SET TIME OUT TRAP TO RETURN
1313 005470 005000          MOV  #RT1,#ERRVEC
1314 005472 005720          CLR  R0
1315 005474 000776          TST  (R0)+
1316 005476 162700 000002 25I  BR   ,+2          ;WILL TIME OUT WHEN END OF MEMORY
1317 005502 010077          SUB  #2,R0
1318 005504 000000          MOV  R0,(PC)+    ;SET VALUE INTO LSTMEM
1319 005506 105737 000771          LSTMEMI WORD 0    ;CONTAINS VALUE OF LAST MEMORY ADDRESS
1320 005512 001003          TSTB #QV         ;NO NEED TO PRESERVE LOADERS
1321 005514 162737 004000 005504 15I  BNE  #4000,#LSTMEM ;IF QV
1322 005522 012737 032372 001012 15I  SUB  #ENDTAG+2,#FRSTMEM ;SET PROTECTION FOR LOADERS
1323 005530 000423          BR   START3      ;SET LOWER BOUNDARY
1324
1325          ;PROGRAM STARTS HERE WHEN ADDRESS 204 IS USED AS STARTING ADDRESS,
1326 005532 012737 002564 000020 START1I MOV  #,TYPE,#IOTVEC ;SET IOT VECTOR TO TYPE ROUTINE
1327 005540 000004 032277          TYPE,MSG1
1328 005544 004567 176474          JSR  R5,RECO
1329 005550 000000 15I  ,WORD 0          ;GET LOWER LIMIT
1330 005552 016737 177772 001012 15I  MOV  ,#FRSTMEM    ;CONTAINS TYPED LOWER LIMIT
1331 005560 000004 032314          TYPE,MSG2          ;SET IN LOWER LIMIT
1332 005564 004567 176454          JSR  R5,RECO
1333 005570 000000 25I  ,WORD 0          ;GET UPPER LIMIT
1334 005572 016737 177772 005504 25I  MOV  #LSTMEM
1335
1336 005600 005037 001000          START3I CLR  #ICNT        ;CLEAR PASS COUNT
1337 005604 105037 000770          CLR  #MHON        ;SET MEM MGMT ON IND=NOT ON
1338 005610 004737 000120          JSR  PC,#,MAHF    ;GO ENABLE PARITY IF AVAILABLE
1339 005614 012737 001600 032006          MOV  #1600,#NEXPAR
1340 005622 012737 020040 001150          MOV  #20040,#ITCNT ;SET TEST ITERATION COUNT
1341 005630 105737 000771          TSTB #QV         ;BRANCH IF NOT IN QV MODE
1342 005634 001403          BEQ  START2
1343 005636 012737 000401 001150          MOV  #403,#ITCNT ;SET 1 ITERATION FOR TESTS
1344
1345          ;PROGRAM RESTARTS HERE AFTER RELOCATION ABOVE 28K IS COMPLETE,
1346 005644 012706 000500          START2I MOV  #STKPTR,SP   ;SET STACK PTR
1347 005650 012737 005274 000004          MOV  #ERRPT,#ERRVEC ;SET ERROR TRAP
1348 005656 012737 005264 000010          MOV  #RESERR,#RESVEC ;SET RESERVED INST TRAP VECTOR
1349 005664 012737 000002 000012          MOV  #RT1,#RESVEC+2
1350 005672 012737 000610 000024          MOV  #PDW,#PFVEC   ;SET POWER FAIL TRAP VECTOR
1351 005700 012737 000340 000026          MOV  #340,#PFVEC+2 ;AND PRIORITY LEVEL 7
1352 005706 012737 005244 000250          MOV  #KTABRT,#MHVEC ;SET KT11 ABORT VECTOR

```

```

1353 005714 012737 002564 000020      MOV    #,TYPE,0,IOTVEC      ;SET IOT VECTOR TO TYPE ROUTINE
1354 005722 012737 002000 000022      MOV    #PRTY4,0,IOTVEC+2    ;SET LEVEL 4 ON TRAP
1355 005730 012737 001014 000030      MOV    #SCOPEA,0,EMTVEC     ;SET EMT(SCOPE) TRAP VECTOR
1356 005736 012737 003212 000034      MOV    #,HLT,0,TRAPVEC     ;SET TRAP (HLT) VECTOR
1357 005744 012737 000200 000036      MOV    #200,0,TRAPVEC+2    ;PRIORITY LEVEL 4 ON TRAP
1358 005752 005037 005322                CLR    #MERFLAG            ;CLEAR ABORT & TRAP TO 4 FLAG
1359 005756 005037 000772                CLR    #DEVID              ;
1360 005762 004737 005166                JSR    PC,0,LODISP         ;LOAD DISPLAY REGISTER
1361 005766 005037 000761                CLR    #PEFLG              ;CLEAR PARITY ERROR FLAG
1362 005772 052737 000100 177560        BIS    #I0B,0,TKS          ;SET IE BIT IN KEYBOARD STATUS REG
1363
1364
1365
1366 226000 105727                ;THE BELOW ROUTINE ASCERTAINS WHICH CP & CP OPTIONS THE PROGRAM IS RUN-
1367 006002 000000                INING ON AND SETS AN INDICATOR IN OPT,CP ACCORDINGLY,
1368 006004 001126                CPCHK) TSTB (PC)+         ;BRANCH IF OPT,CP HAS BEEN TYPED
1369 006006 012737 000006 000004        BNE    0                   ;
1370 006014 012737 000012 000010        MOV    #ERRVEC+2,0,ERRVEC  ;SET UP ERROR TRAP TO RETURN
1371 006022 012700 000004                MOV    #RESVEC+2,0,RESVEC  ;AND ALSO RESERVED INST TRAP
1372 006026 000261                SEC
1373 006030 005037 177766                CLR    #CPUERR             ;CLEAR CPU ERROR REG
1374 006034 005600                SBC    R0
1375 006036 000261                SEC
1376 006040 005737 177772                TST    #PIRQ               ;R0=3 IF 11/45
1377 006044 005600                SBC    R0                  ;R0=2 IF 11/40
1378 006046 000261                SEC
1379 006050 105737 177777                TSTB   #PSW+1              ;TIMES OUT IF 11/20
1380 006054 005600                SBC    R0                  ;R0=1 IF 11/20
1381 006056 005037 177700                CLR    #I17700            ;CLEARS R0 IF 11/05
1382 006062 006300                ASL    R0                  ;SHIFT CP INDICATOR
1383 006064 010002                MOV    R0,R2              ;MOVE CP TYPE TO R2
1384 006066 000261                SEC
1385 006070 005737 177572                TST    #SR0                ;CHECK IF MEM MGMT IS AVAILBLE
1386 006074 103402                BCS   15                   ;
1387 006076 052702 100000                BIS    #TOPT,R2           ;SET MEM MGMT AVAIL INDICATOR
1388 006102 005004                CLR    R4
1389 006104 000261                SEC
1390 006106 072404                ASH    R4,R4              ;WILL TRAP IF 11/40 WITHOUT EIS
1391 006110 103402                BCS   25                   ;BRANCH IF NO EIS AVAILABLE
1392 006112 052702 040000                BIS    #EISOPT,R2        ;SET EIS AVAIL INDICATOR
1393 006116 000261                SEC
1394 006120 170500                TSTB   R0                  ;SET CARRY
1395 006122 170000                CFCC
1396 006124 103402                BCS   35                   ;WILL CLEAR CARRY IF 11/45 FLOATING POINT
1397 006126 052702 020000                BIS    #FOPT,R2          ;IS AVAIL, COPY FLGATING CC'S INTO PSW
1398 006132 000261                SEC
1399 006134 075000                FADD   R0                  ;BRANCH IF NO FLOATING POINT
1400 006136 103402                SBC    R0                  ;SET FP OPTION AVAIL INDICATOR
1401 006140 052702 010000                BIS    #FISOPT,R2        ;SET 'C' BIT
1402 006144 000261                SEC
1403 006146 005037 177774                CLR    #SLR               ;BRANCH IF NO FIS OPTION
1404 006152 103402                BCS   45                   ;SET FIS OPTION AVAIL INDICATOR
1405 006154 052702 004000                BIS    #KJQPT,R2         ;SET 'C' BIT
1406 006160 000261                SEC
1407 006162 005737 172540                TST    #PLKCR             ;BRANCH IF NO KW11-P
1408 006166 103402                BCS   65                   ;
1409 006170 052702 002000                BIS    #PLKOPT,R2        ;SET OPTION INDICATOR
1410 006174 000261                SEC
1411 006176 005737 177546                TST    #LKS               ;BRANCH IF NO KW11-L
1412 006202 103402                BCS   75                   ;
1413 006204 052702 001000                BIS    #LKOPT,R2         ;SET OPTION INDICATOR
1414 006210 005737 177564                TST    #TFS               ;BRANCH IF NO CONSOLE TTY
1415 006214 103402                BCS   85                   ;
1416 006216 052702 000400                BIS    #TOPT,R2          ;
1417 006222 012737 005274 000004        MOV    #ERPRT,0,ERRVEC    ;RESTORE ERROR TRAP
1418 006230 012737 005264 000010        MOV    #RESERR,0,RESVEC   ;AND ALSO RESERVED INST TRAP
1419 006236 010237 000764                MOV    R2,0,POPT,CP      ;LOAD INDICATOR
1420 006242 000004 032264                TYPE, AOPT, CP
1421 006246 004767 174632                JSR    PC, VYPDAT
1422 006252 000004 000752                TYPE, CRUF
1423 006256 105267 177520                INCB   105
1424 006262
  
```

```

1407 006162 005737 172540                TST    #PLKCR             ;BRANCH IF NO KW11-P
1408 006166 103402                BCS   65                   ;
1409 006170 052702 002000                BIS    #PLKOPT,R2        ;SET OPTION INDICATOR
1410 006174 000261                SEC
1411 006176 005737 177546                TST    #LKS               ;BRANCH IF NO KW11-L
1412 006202 103402                BCS   75                   ;
1413 006204 052702 001000                BIS    #LKOPT,R2         ;SET OPTION INDICATOR
1414 006210 005737 177564                TST    #TFS               ;BRANCH IF NO CONSOLE TTY
1415 006214 103402                BCS   85                   ;
1416 006216 052702 000400                BIS    #TOPT,R2          ;
1417 006222 012737 005274 000004        MOV    #ERPRT,0,ERRVEC    ;RESTORE ERROR TRAP
1418 006230 012737 005264 000010        MOV    #RESERR,0,RESVEC   ;AND ALSO RESERVED INST TRAP
1419 006236 010237 000764                MOV    R2,0,POPT,CP      ;LOAD INDICATOR
1420 006242 000004 032264                TYPE, AOPT, CP
1421 006246 004767 174632                JSR    PC, VYPDAT
1422 006252 000004 000752                TYPE, CRUF
1423 006256 105267 177520                INCB   105
1424 006262
  
```

1425						;;SBTTL	START OF SECTION 0
1426						10000000000000	FIRST ADDRESS TO BE RELOCATED 00000000
1427						RELOC	MOV PC,R0 ;L PC
1428	006262	010700				;(R0)	IRI CONTAINS THE ADDRESS OF REL0
1429	006264	005740				MOV R0,#PRSTAD	ISAVE
1430	006266	010037	001010			MOV #0,#SECT	ISSET SECTION #
1431	006272	012737	000000	005176		JSR PC,#LDISPL	ILoad DISPLAY REG
1432	006300	004737	005166			MOV @#DISPL,REL00	
1433	006304	013767	005172	000740		MOV PC,R0	IGET CURRENT PC
1434	006312	010700				SUB #,R0	ISUBTRACT RELOCATION FACTOR
1435	006314	162700	006314			MOV R0,#FACTOR	ISAVE RELOCATION FACTOR
1436	006320	010037	001004			MOV PC,R1	ISSET NEW SCOPE PTR
1437	006324	010701				INMEMORY AND DISK	(IK SELECTED) VERIFICATION TEST,
1438						JMP	15
1439	006326	000167	000714			.WORD	=1,-1,-1,-1,0,0,0,0
1440	006332	177777	177777	177777			
1441	006340	177777	000000	000000			
1442	006346	000000	000000				
1443	006352	177777	177777	177777		.WORD	=1,-1,-1,-1,0,0,0,0
1444	006360	177777	000000	000000			
1445	006366	000000	000000				
1446	006372	177777	177777	177777		.WORD	=1,-1,-1,-1,0,0,0,0
1447	006400	177777	000000	000000			
1448	006406	000000	000000				
1449	006412	177777	177777	177777		.WORD	=1,-1,-1,-1,0,0,0,0
1450	006420	177777	000000	000000			
1451	006426	000000	000000				
1452	006432	177777	177777	177777		.WORD	=1,-1,-1,-1,0,0,0,0
1453	006440	177777	000000	000000			
1454	006446	000000	000000				
1455	006452	177777	177777	177777		.WORD	=1,-1,-1,-1,0,0,0,0
1456	006460	177777	000000	006400			
1457	006466	000000	000000				
1458	006472	177777	177777	177777		.WORD	=1,-1,-1,-1,0,0,0,0
1459	006500	177777	000000	000000			
1460	006506	000000	000000				
1461	006512	177777	177777	177777		.WORD	=1,-1,-1,-1,0,0,0,0
1462	006520	177777	000000	000000			
1463	006526	000000	000000				
1464	006532	177777	177777	177777		.WORD	=1,-1,-1,-1,0,0,0,0
1465	006540	177777	000000	000000			
1466	006546	000000	000000				
1467	006552	177777	177777	177777		.WORD	=1,-1,-1,-1,0,0,0,0
1468	006560	177777	000000	000000			
1469	006566	000000	000000				
1470	006572	177777	177777	177777		.WORD	=1,-1,-1,-1,0,0,0,0
1471	006600	177777	000000	000000			
1472	006606	000000	000000				
1473	006612	177777	177777	177777		.WORD	=1,-1,-1,-1,0,0,0,0
1474	006620	177777	000000	000000			
1475	006626	000000	000000				
1476	006632	177777	177777	177777		.WORD	=1,-1,-1,-1,0,0,0,0
1477	006640	177777	000000	000000			
1478	006646	000000	000000				

1479	006652	177777	177777	177777		.WORD	=1,-1,-1,-1,0,0,0,0
1480	006660	177777	000000	000000			
1481	006666	000000	000000				
1482	006672	177777	177777	177777		.WORD	=1,-1,-1,-1,0,0,0,0
1483	006700	177777	000000	000000			
1484	006706	000000	000000				
1485	006712	177777	177777	177777		.WORD	=1,-1,-1,-1,0,0,0,0
1486	006720	177777	000000	000000			
1487	006726	000000	000000				
1488	006732	177777	177777	177777		.WORD	=1,-1,-1,-1,0,0,0,0
1489	006740	177777	000000	000000			
1490	006746	000000	000000				
1491	006752	177777	177777	177777		.WORD	=1,-1,-1,-1,0,0,0,0
1492	006760	177777	000000	000000			
1493	006766	000000	000000				
1494	006772	177777	177777	177777		.WORD	=1,-1,-1,-1,0,0,0,0
1495	007000	177777	000000	000000			
1496	007006	000000	000000				
1497	007012	177777	177777	177777		.WORD	=1,-1,-1,-1,0,0,0,0
1498	007020	177777	000000	000000			
1499	007026	000000	000000				
1500	007032	177777	177777	177777		.WORD	=1,-1,-1,-1,0,0,0,0
1501	007040	177777	000000	000000			
1502	007046	000000	000000				
1503	007052	177777	177777	177777		.WORD	=1,-1,-1,-1,0,0,0,0
1504	007060	177777	000000	000000			
1505	007066	000000	000000				
1506	007072	177777	177777	177777		.WORD	=1,-1,-1,-1,0,0,0,0
1507	007100	177777	000000	000000			
1508	007106	000000	000000				
1509	007112	177777	177777	177777		.WORD	=1,-1,-1,-1,0,0,0,0
1510	007120	177777	000000	000000			
1511	007126	000000	000000				
1512	007132	177777	177777	177777		.WORD	=1,-1,-1,-1,0,0,0,0
1513	007140	177777	000000	000000			
1514	007146	000000	000000				
1515	007152	177777	177777	177777		.WORD	=1,-1,-1,-1,0,0,0,0
1516	007160	177777	000000	000000			
1517	007166	000000	000000				
1518	007172	177777	177777	177777		.WORD	=1,-1,-1,-1,0,0,0,0
1519	007200	177777	000000	000000			
1520	007206	000000	000000				
1521	007212	177777	177777	177777		.WORD	=1,-1,-1,-1,0,0,0,0
1522	007220	177777	000000	000000			
1523	007226	000000	000000				
1524	007232	177777	177777	177777		.WORD	=1,-1,-1,-1,0,0,0,0
1525	007240	177777	000000	000000			
1526	007246					151	
1527	007246	010702				MOV	PC,R2
1528	007250	062702	000012			ADD	#12,R2
1529	007254	012707	001152			MOV	#RELOC,PC
1530	007260	000000				IGO	RELOCATE PROGRAM CODE
1531						REL00	0
1532						10000000000000	LAST ADDRESS OF CODE TO BE RELOCATED 0000000000

1533										
1534										
1535										
1536	007262	010700								
1537	007264	005740								
1538	007266	010037	001010							
1539	007272	012737	000001	005176						
1540	007300	024737	005166							
1541	007304	013767	005172	003776						
1542	007312	010700								
1543	007314	162700	007314							
1544	007320	010037	001004							
1545	007324	010701								
1546										
1547	007326	000257								
1548	007330	103407								
1549	007332	102406								
1550	007334	001405								
1551	007336	100404								
1552	007340	002403								
1553	007342	003402								
1554	007344	101401								
1555	007346	101001								
1556	007350	104400								
1557										
1558										
1559	007352	000270								
1560	007354	100003								
1561	007356	002002								
1562	007360	003001								
1563	007362	002401								
1564	007364	104400								
1565										
1566										
1567	007366	000262								
1568	007370	102003								
1569	007372	002402								
1570	007374	003401								
1571	007376	002001								
1572	007400	104400								
1573										
1574										
1575	007402	000261								
1576	007404	103002								
1577	007406	101001								
1578	007410	003001								
1579	007412	104400								
1580										
1581										
1582	007414	000264								
1583	007416	001003								
1584	007420	003002								
1585	007422	101001								
1586	007424	003401								

1587	007426	104400								
1588	007430	104000								
1589										
1590										
1591										
1592	007432	000277								
1593	007434	000244								
1594	007436	005000								
1595	007440	103404								
1596	007442	102403								
1597	007444	001002								
1598	007446	100401								
1599	007450	003401								
1600	007452	104400								
1601										
1602	007454	000277								
1603	007456	000244								
1604	007460	005700								
1605	007462	103404								
1606	007464	102403								
1607	007466	001002								
1608	007470	100401								
1609	007472	101401								
1610	007474	104400								
1611										
1612	007476	000257								
1613	007500	000266								
1614	007502	005100								
1615	007504	102403								
1616	007506	102403								
1617	007510	001402								
1618	007512	100001								
1619	007514	002401								
1620	007516	104400								
1621										
1622	007520	000261								
1623	007522	005000								
1624	007524	103003								
1625	007526	102402								
1626	007530	001001								
1627	007532	002001								
1628	007534	104400								
1629										
1630	007536	000261								
1631	007540	006000								
1632	007542	103404								
1633	007544	102003								
1634	007546	001402								
1635	007550	000001								
1636	007552	003001								
1637	007554	104400								
1638	007556	000277								
1639	007560	000242								
1640	007562	005300								

1641	007564	103004	BCC	DEC0	
1642	007566	102003	BVC	DEC0	
1643	007570	031402	BEQ	DEC0	
1644	007572	100401	BMI	DEC0	
1645	007574	003401	BLE	,*4	
1646	007576	104400	DEC0	HLT	ERROR! INCORRECT CC'S AFTER DEC
1647					
1648	007600	000257	CCC		
1649	007602	005200	INC	R0	IR0=100000,CC'S=1010
1650	007604	103404	BCC	INC0	
1651	007606	102003	BVC	INC0	
1652	007610	001402	BEQ	INC0	
1653	007612	100001	BPL	INC0	
1654	007614	003001	BGT	,*4	
1655	007616	104400	INC0	HLT	ERROR! INCORRECT CC'S AFTER INC
1656					
1657	007620	000277	SCC		
1658	007622	000242	CLV		
1659	007624	005400	NE0	R0	IR0=100000,CC'S=1011
1660	007626	103003	BCC	NEG0	
1661	007630	102002	BVC	NEG0	
1662	007632	001401	BEQ	NEG0	
1663	007634	002001	BGE	,*4	
1664	007636	104400	NEG0	HLT	ERROR! INCORRECT CC'S AFTER NEG
1665					
1666	007640	000261	SEC		
1667	007642	004300	ASL	R0	IR0=000000,CC'S=0111
1668	007644	103004	BCC	ASL0	
1669	007646	102003	BVC	ASL0	
1670	007650	001002	BNE	ASL0	
1671	007652	100401	BMI	ASL0	
1672	007654	101401	BLOS	,*4	
1673	007656	104400	ASL0	HLT	ERROR! INCORRECT CC'S AFTER ASL
1674					
1675	007660	006100	ROL	R0	IR0=000001,CC'S=0000
1676	007662	103402	BCC	ROL0	
1677	007664	003401	BLE	ROL0	
1678	007666	002001	BGE	,*4	
1679	007670	104400	ROL0	HLT	ERROR! INCORRECT CC'S AFTER ROL
1680					
1681	007672	006200	ASR	R0	IR0=000000,CC'S=0111
1682	007674	103003	BCC	ASR0	
1683	007676	102002	BVC	ASR0	
1684	007700	001001	BNE	ASR0	
1685	007702	002401	BLT	,*4	
1686	007704	104400	ASR0	HLT	ERROR! INCORRECT CC'S AFTER ASR
1687					
1688	007706	000277	SCC		
1689	007710	005600	SBC	R0	IR0=1,CC'S=1001
1690	007712	103002	BCC	SBC0	
1691	007714	102401	BVS	SBC0	
1692	007716	003401	BLE	,*4	
1693	007720	104400	SBC0	HLT	ERROR! INCORRECT CC'S AFTER SBC
1694					

1695	007722	005400	NEG	R0	IR0=000001,CC'S=00001
1696	007724	000300	SWAB	R0	IR0=000400,CC'S=0100
1697	007726	103403	BCC	SWAB0	
1698	007730	102402	BVS	SWAB0	
1699	007732	001001	BNE	SWAB0	
1700	007734	002001	BGE	,*4	
1701	007736	104400	SWAB0	HLT	ERROR! INCORRECT CC'S AFTER SWAB
1702	007740	104000	SCOPE		
1703					
1704			ICHECK REGISTER SELECTION		
1705	007742	005000	CLR	R0	
1706	007744	000277	SCC		
1707	007746	006100	ROL	R0	IR0=1
1708	007750	010002	MOV	R0,R2	
1709	007752	006302	ASL	R2	IR2=2
1710	007754	010203	MOV	R2,R3	
1711	007756	006303	ASL	R3	IR3=4
1712	007760	010304	MOV	R3,R4	
1713	007762	006304	ASL	R4	IR4=10
1714	007764	010405	MOV	R4,R5	
1715	007766	006305	ASL	R5	IR5=20
1716	007770	010546	MOV	R5,(SP)	IR5=20
1717	007772	004116	BIS	R4,(SP)	IR5=20
1718	007774	005316	BIS	R3,(SP)	IR5=20
1719	007776	005216	BIS	R2,(SP)	IR5=20
1720	010000	005016	BIS	R0,(SP)	IR5=20
1721	010002	022726	000037	CHP	#37,(SP)*
1722	010006	001401	BEQ	,*4	
1723	010010	104400	HLT		WHERE SET
1724					MISSING BIT(S) REPRESENT
1725					INCORRECT REGISTER SELECTION
1726					
1727	010012	000257	CCC		
1728	010014	112700	000377	MOV	#37,R0
1729	010020	006100	15	ROL	R0
1730	010022	103776	BCC	15	IR0TATE A 0 THROUGH ALL BIT
1731	010024	005200	INC	R0	POSITIONS
1732	010026	001401	BEQ	,*4	IF FINAL RESULT IS 1
1733	010030	104400	HLT		ERROR!
1734					
1735	010032	012700	000020	MOV	#16,R0
1736	010036	005002	CLR	R2	IR0TATE 1 THROUGH ALL BIT POSITS
1737	010040	000261	SEC		DECREMENT SHIFT COUNT
1738	010042	006002	ROR	R2	
1739	010044	005300	DEC	R0	IR2 SHOULD CONTAIN -1
1740	010046	001374	BNE	25	
1741	010050	005102	COM	R2	
1742	010052	001401	BEQ	,*4	
1743	010054	104400	HLT		ERROR! CHECK R2 SHOULD = 0
1744					
1745	010056	012703	100000	MOV	#100000,R3
1746	010062	006203	35	ASR	R3
1747	010064	103376	BCC	35	
1748	010066	005203	INC	R3	IR0TATE 1 BIT THROUGH ALL POSITIONS

1749	010070	001401		BEQ	,+4	
1750	010072	104400		HLT		!ERROR!
1751						
1752	010074	112704	177401	MOV#B	#177401,R4	!R4=1
1753	010100	000404		ADD	R4,R4	!HAS THE AFFECT OF SHIFTING A BIT
1754	010102	103376		BCC	45	!THROUGH ALL POSITIONS
1755	010104	005704		TST	R4	!RESULT SHOULD BE 0
1756	010106	001401		BEQ	,+4	
1757	010110	104400		HLT		
1758						
1759	010112	012705	000001	MOV	#1,R5	
1760	010116	006305		ASL	R5	
1761	010120	102376		BVC	55	
1762	010122	006305		ASL	R5	
1763	010124	103002		RCC	65	
1764	010126	005705		TST	R5	
1765	010130	001401		BEQ	,+4	
1766	010132	104400		HLT		
1767						
1768						
1769	010134	005002		!CHECK REGISTER VOLATILITY		
1770	010136	005102		CLR	R2	
1771	010140	010203		COM	R2	!R2=1
1772	010142	000257		MOV	R2,R3	
1773	010144	006002		CCC		
1774	010146	006202		ROR	R2	!R2=LOOP COUNT
1775	010150	010304		ASR	R2	
1776	010152	005302		MOV	R3,R4	
1777	010154	001375		DEC	R2	!DECREMENT LOOP COUNT
1778	010156	005203		BNE	75	
1779	010160	001002		INC	R3	!CHECK R3
1780	010162	005204		BNE	85	
1781	010164	001401		INC	R4	!CHECK R4
1782	010166	104400		BEQ	,+4	
1783				HLT		
1784						
1785	010170	012737	000001 177776	!CHECK TRANSFER OF REGISTER DATA BETWEEN THE GS AND GD REGISTERS (11/45)		
1786	010176	001052		GSTST	BIT	!CHECK IF 'T' BIT IS SET
1787	010200	010146		BNE	75	!SKIP TEST IF 'T' BIT SET
1788	010202	010627		MOV	R1,=(SP)	!SAVE SCOPE PTR
1789	010204	000000		MOV	SP,=(PC)+	!SAVE STACK PTR
1790	010206	010727		!WORD	0	!CONTAINS SAVED STACK PTR
1791	010210	000000		MOV	PC,=(PC)+	!LOAD DATA, THE CURRENT PC IS USED AS
1792				!WORD	0	!DATA, IF THIS TEST FAILS 2'S CON-
1793	010212	005267		INC	25	!TAINS THE DATA BEING USED,
1794	010216	016700	177772	MOV	25,R0	!MAKE ODD TO CHECK BIT 0
1795	010222	010001	177766	MOV	R0,R1	!LOAD GD REGISTER 0
1796	010224	010103		MOV	R1,R2	!TRANSFER GS REG 0 TO GD REG 1
1797	010226	010203		MOV	R2,R3	!AND GS REG 1 TO GD REG 2
1798	010230	010304		MOV	R3,R4	!ETC,...
1799	010232	010405		MOV	R4,R5	
1800	010234	152737	000340 177776	BISB	#340,00PSW	!SET PRIORITY LEVEL 7
1801	010242	010506		MOV	R5,SP	!TRANSFER GS REG 5 TO GD STK PTR
1802	010244	010627		MOV	SP,=(PC)+	!TRANSFER GS STK PTR TO MEMORY

1803	010246	000000		45:	!WORD	0	!CONTAINS GS STACK PTR
1804	010250	016700	177730	MOV	15,SP		!RESTORE STK PTR NEEDED FOR HLT/SCOPE
1805	010254	142737	000340 177776	BICB	#340,00PSW		!SET PRIORITY LEVEL 0
1806	010262	026700	177760	CM#	45,R0		!COMPARE GS/GD STKPTR WITH GS REG 0
1807	010266	001004		BNE	55		!BRANCH IF THEY WERE NOT =
1808	010270	006307	177714	ASL	25		!SHIFT TEST DATA UNTIL = 000000
1809	010274	001350		BNE	35		
1810	010276	000411		BR	65		
1811	010300	010046		55:	MOV	R0,=(SP)	!GET GS REG 0
1812	010302	010146		MOV	R1,=(SP)		!ETC,...
1813	010304	010246		MOV	R2,=(SP)		
1814	010306	010346		MOV	R3,=(SP)		
1815	010310	010446		MOV	R4,=(SP)		
1816	010312	010546		MOV	R5,=(SP)		
1817	010314	104400		HLT			
1818							
1819	010316	016700	177662	MOV	15,SP		!ERROR! DATA IN GS STK PTR NOT = GS REG 0
1820	010322	012601		65:	MOV	=(SP)+,R1	!GS REG 0=GS REG 5 ARE ON THE STACK
1821	010324	104000		75:	SCOPE		!RESTORE STACK PTR
1822							!RESTORE SCOPE PTR
1823							
1824	010326	000401		!TEST UNARY WORD INSTRUCTIONS USING ADDRESS MODE 1			
1825	010330	000000		BR	,+4		
1826	010332	010702		!WORD	0		!RESERVE ADDRESS FOR TESTS
1827	010334	102702	000004	MOV	PC,R2		
1828	010340	005012		SUB	#4,R2		!R2 POINTS TO RESERVED WORD
1829				CLR	(R2)		!PRESET (R2)
1830	010342	000261		SEC			
1831	010344	000012		ROR	(R2)		!(R2)=100000,CC=1010
1832	010346	101402		BLOS	R0,R1		
1833	010350	100001		BPL	R0,R1		
1834	010352	002001		BGE	,+4		
1835	010354	104400		HLT			!ERROR! INCORRECT CC'S AS SHOWN ABOVE
1836							
1837	010356	000257		CCC			
1838	010360	000241		SEC			
1839	010362	005312		DEC	(R2)		!(R2)=077777,CC=0011
1840	010364	103001		BCC	DEC1		
1841	010366	003401		BLE	,+4		
1842	010370	104400		DEC1:	HLT		!ERROR! INCORRECT CC'S AS SHOWN ABOVE
1843							
1844	010372	000257		CCC			
1845	010374	000261		SEC			
1846	010376	005512		ADC	(R2)		!(R2)=100000,CC=1010
1847	010400	103403		BCS	ADC1		
1848	010402	102002		BVC	ADC1		
1849	010404	100001		BPL	ADC1		
1850	010406	001001		BNE	,+4		
1851	010410	104400		ADC1:	HLT		!ERROR! INCORRECT CC'S AS SHOWN ABOVE
1852							
1853	010412	006112		ROL	(R2)		!(R2)=000000,CC=0111
1854	010414	103003		RCC	ROL1		
1855	010416	102002		RVC	ROL1		
1856	010420	001001		BNE	ROL1		

1857	010422	100001			
1858	010424	104400	ROL1	BPL HLT	,+4 ERROR! INCORRECT CC'S AS SHOWN ABOVE
1859					
1860	010426	006112		ROL (R2)	,(R2)=000001,CC=0000
1861	010430	101402		BLOS R01,4	BRANCH IF C OR Z IS SET
1862	010432	102401		BVS ROL1A	
1863	010434	100001		BPL ,+4	
1864	010436	104400	ROL1A1	HLT	
1865					
1866	010440	006212		ASR (R2)	,(R2)=000000,CC=0111
1867	010442	103003		BCC ASR1	
1868	010444	102002		BVC ASR1	
1869	010446	001001		BNE ASR1	
1870	010450	100001		BPL ,+4	
1871	010452	104400	ASR1	HLT	ERROR! INCORRECT CC'S AS SHOWN ABOVE
1872					
1873	010454	006012		ROR (R2)	,(R2)=100000,CC=1010
1874	010456	103403		ROR1A ROR1A	
1875	010460	102002		BVC ROR1A	
1876	010462	001401		BEO ROR1A	
1877	010464	100401		BMI ,+4	
1878	010466	104400	ROR1A1	HLT	
1879					
1880	010470	000261		SEC	
1881	010472	005212		INC (R2)	,(R2)=100001,CC=1001
1882	010474	103003		BCC INC1	
1883	010476	102402		BVS INC1	
1884	010500	001401		BEO INC1	
1885	010502	100401		BMI ,+4	
1886	010504	104400	INC1	HLT	ERROR! INCORRECT CC'S AS SHOWN ABOVE
1887					
1888	010506	005612		SBC (R2)	,(R2)=100000,CC=1000
1889	010510	103403		SBC1 SBC1	
1890	010512	102402		BVS SBC1	
1891	010514	001401		BEO SBC1	
1892	010516	100401		BMI ,+4	
1893	010520	104400	SBC1	HLT	ERROR! INCORRECT CC'S AS SHOWN ABOVE
1894					
1895	010522	000261		SEC	
1896	010524	005612		SBC (R2)	,(R2)=077777,CC=0010
1897	010526	103403		SBC1A SBC1A	
1898	010530	102002		BVC SBC1A	
1899	010532	001401		BEO SBC1A	
1900	010534	100001		BPL ,+4	
1901	010536	104400	SBC1A1	HLT	ERROR! INCORRECT CC'S AS SHOWN ABOVE
1902					
1903	010540	000261		SEC	
1904	010542	005512		ADC (R2)	,(R2)=100000,CC=1010
1905	010544	100401		BMI ,+4	
1906	010546	104400		HLT	
1907					
1908	010550	000261		SEC	
1909	010552	006312		ASL (R2)	,(R2)=000000,CC=0111
1910	010554	103003		BCC ASL1	

1911	010556	102002		BVC ASL1	
1912	010560	001001		BNE ASL1	
1913	010562	100001		BPL ,+4	
1914	010564	104400	ASL1	HLT	ERROR! INCORRECT CC'S AS SHOWN ABOVE
1915					
1916	010566	005112		COM (R2)	,(R2)=177777,CC=1001
1917	010570	103002		BCC COM1	
1918	010572	102401		BVS COM1	
1919	010574	100401		BMI ,+4	
1920	010576	104400	COM1	HLT	ERROR! INCORRECT CC'S AS SHOWN ABOVE
1921					
1922	010600	000250		CLN	
1923	010602	005712		TST (R2)	,(R2)=177777,CC=1000
1924	010604	103403		BCC TST1	
1925	010606	102402		BVS TST1	
1926	010610	100001		BPL TST1	
1927	010612	001001		BNE ,+4	
1928	010614	104400	TST1	HLT	ERROR! INCORRECT CC'S AS SHOWN ABOVE
1929					
1930	010616	000262		SEV	
1931	010620	005412		NEG (R2)	,(R2)=000001,CC=0000
1932	010622	103002		BCC NEG1	
1933	010624	102401		BVS NEG1	
1934	010626	001001		BNE	
1935	010630	104400	NEG1	HLT	ERROR! INCORRECT CC'S AS SHOWN ABOVE
1936					
1937	010632	005312		DEC (R2)	,(R2)=000000,CC=0101
1938	010634	103001		BCC DEC1A	
1939	010636	001401		BEO ,+4	
1940	010640	104400	DEC1A1	HLT	ERROR! INCORRECT CC'S AS SHOWN ABOVE
1941	010642	104000		SCOPE	
1942					
1943					
1944	010644	000401		JCHECK UNARY BYTE INSTRUCTIONS USING ADDRESS MODE 1	
1945	010646	000000		BR ,+4	RESERVE A WORD
1946	010650	010703		WORD 0	ADDRESS RESERVED FOR TESTS
1947	010652	162703	000004	MOV PC,R3	
1948	010656	010304		SUB #4,R3	R3 POINTS TO EVEN BYTE OF WORD
1949	010660	005204		MOV R3,R4	R4 POINTS TO ODD BYTE OF WORD
1950	010662	005013		INC R4	
1951				CLR (R3)	PRESET DATA
1952	010664	000261	1\$1	SEC	
1953	010666	105513		ADCB (R3)	,ADD CARRY TO EVEN BYTE
1954	010670	100402		BMI 2\$	UNTIL EVEN BYTE BECOMES NEGATIVE
1955	010672	105214		INCB (R4)	INCREMENT ODD BYTE
1956	010674	000773		BR 1\$	
1957	010676	102401	2\$1	BVS ,+4	,(R3)=077600=[0774][200],CC=1010
1958	010700	104400		HLT	
1959	010702	000242		CLV	
1960	010704	105214		INCB (R4)	,(R3)=100200=[1000][200],CC=1010
1961	010706	103402		RCS INCB1	
1962	010710	102001		RVC INCB1	
1963	010712	100401		BMI ,+4	
1964	010714	104400	INCB1	HLT	ERROR! INCORRECT CC'S AS SHOWN ABOVE

1965					
1966	010716	106114	ROLB	(R4)	
1967	010720	103002	RCC	ROLB1	
1968	010722	102001	RVC	ROLB1	
1969	010724	001401	BEO	,+4	
1970	010726	104400	ROLB1	HLT	ERROR! INCORRECT CC'S AS SHOWN ABOVE
1971					
1972	010730	105614	SDCB	(R4)	
1973	010732	103002	RCC	SBCB1	
1974	010734	102401	BVS	SBCB1	
1975	010736	004021	BMI	,+4	
1976	010740	104400	SBCB1	HLT	ERROR! INCORRECT CC'S AS SHOWN ABOVE
1977					
1978	010742	106313	ASLB	(R3)	
1979	010744	103002	RCC	ASLB1	
1980	010746	102001	RVC	ASLB1	
1981	010750	001401	REQ	,+4	
1982	010752	104400	ASLB1	HLT	ERROR! INCORRECT CC'S AS SHOWN ABOVE
1983					
1984	010754	105413	NECB	(R3)	
1985	010756	103402	RCS	NECB1	
1986	010760	102401	BVS	NECB1	
1987	010762	001401	BEQ	,+4	
1988	010764	104400	NECB1	HLT	ERROR! INCORRECT CC'S AS SHOWN ABOVE
1989					
1990	010766	000277	SCC		
1991	010770	105313	DECB	(R3)	
1992	010772	103002	RCC	DECB1	
1993	010774	102401	BVS	DECB1	
1994	010776	001001	BNE	,+4	
1995	011000	104400	DECB1	HLT	ERROR! INCORRECT CC'S AS SHOWN ABOVE
1996					
1997	011002	000241	CLC		
1998	011004	106013	RORB	(R3)	
1999	011006	103002	RCC	RORB1	
2000	011010	102001	RVC	RORB1	
2001	011012	000001	BPL	,+4	
2002	011014	104400	RORB1	HLT	ERROR! INCORRECT CC'S AS SHOWN ABOVE
2003					
2004	011016	000241	CLC		
2005	011020	105114	COMB	(R4)	
2006	011022	103002	RCC	COMB1	
2007	011024	102401	BVS	COMB1	
2008	011026	001401	BEQ	,+4	
2009	011030	104400	COMB1	HLT	ERROR! INCORRECT CC'S AS SHOWN ABOVE
2010					
2011	011032	106213	15:	ASRB	(R3)
2012	011034	102002	RVC	25	
2013	011036	105514	ADCB	(R4)	
2014	011040	000774	BR	13	
2015	011042	103401	25:	BVS	ASRB1
2016	011044	001401	BEQ	,+4	
2017	011046	104400	ASRB1	HLT	ERROR! INCORRECT CC'S AS SHOWN ABOVE
2018					

2019	011050	106214	ASRB	(R4)	
2020	011052	106214	ASRB	(R4)	
2021	011054	103002	BCC	ASRB1A	
2022	011056	102001	RVC	ASRB1A	
2023	011060	001001	BNE	,+4	
2024	011062	104400	ASRB1A	HLT	ERROR! INCORRECT CC'S AS SHOWN ABOVE
2025					
2026	011064	105314	DECB	(R4)	
2027	011066	001401	BEQ	,+4	
2028	011070	104400	HLT		ERROR! INCORRECT CC'S AS SHOWN ABOVE
2029					
2030	011072	000261	SEC		
2031	011074	106014	RORB	(R4)	
2032	011076	103402	RCS	RORB1A	
2033	011080	102001	RVC	RORB1A	
2034	011082	100401	BMI	,+4	
2035	011084	104400	RORB1A	HLT	ERROR! INCORRECT CC'S AS SHOWN ABOVE
2036					
2037	011086	000242	CLV		
2038	011088	105314	DECB	(R4)	
2039	011092	102401	BVS	,+4	
2040	011094	104400	HLT		
2041					
2042	011096	000261	SEC		
2043	011098	105313	NECB	(R3)	
2044	011102	103002	RCC	DECB1A	
2045	011104	102401	BVS	DECB1A	
2046	011106	100401	BMI	,+4	
2047	011108	104400	DECB1A	HLT	ERROR! INCORRECT CC'S AS SHOWN ABOVE
2048					
2049	011132	000277	SCC		
2050	011134	000313	SWAB	(R3)	
2051	011136	103402	RCS	SWAB1	
2052	011140	102401	BVS	SWAB1	
2053	011142	100001	BPL	,+4	
2054	011144	104400	SWAB1	HLT	ERROR! INCORRECT CC'S AS SHOWN ABOVE
2055					
2056	011146	105714	TSTB	(R4)	
2057	011150	103402	RCS	TSTB1	
2058	011152	102401	BVS	TSTB1	
2059	011154	100401	BMI	,+4	
2060	011156	104400	TSTB1	HLT	ERROR! INCORRECT CC'S AS SHOWN ABOVE
2061					
2062	011160	105014	CLRB	(R4)	
2063	011162	001401	BEO	,+4	
2064	011164	104400	HLT		
2065	011166	106313	ASLB	(R3)	
2066	011170	103402	RCS	ASLB1A	
2067	011172	102001	RVC	ASLB1A	
2068	011174	100401	BMI	,+4	
2069	011176	104400	ASLB1A	HLT	ERROR! INCORRECT CC'S AS SHOWN ABOVE
2070					
2071	011200	105113	COMB	(R3)	
2072	011202	103002	RCC	COMB1A	

2073	011204	102401		BVS	COMB1A	
2074	011206	100001		BPL	,+4	
2075	011210	104400		COMB1A1	HLT	!ERROR! INCORRECT CC'S AS SHOWN ABOVE
2076						
2077	011212	000313		SWAB	(R3)	!(R3)=000400, CC=0100
2078	011214	001401		BEQ	,+4	
2079	011216	104400		HLT		
2080						
2081	011220	105213		INCB	(R3)	
2082	011222	000261		SEC		
2083	011224	105613		SBCB	(R3)	!(R3)=000400, CC=0100
2084	011226	001401		BEQ	,+4	
2085	011230	104400		HLT		
2086	011232	022713	000400	CHP	#400,(R3)	!CHECK REMAINING RESULT
2087	011236	001401		BEQ	,+4	
2088	011240	104400		HLT		
2089	011242	104000		SCOPE		
2090						
2091						
2092	011244	000401				!CHECK UNARY WORD OPS USING ADDRESS MODES 2 AND 4 (AUTO INC/DEC)
2093	011246	000000		BR	,+4	
2094	011250	010704		WORD	0	!ADDRESS RESERVED FOR TESTS
2095	011252	162704	000004	MOV	PC,R4	
2096	011256	010405		SUB	#4,R4	!R4 AND R5 POINT TO
2097	011260	005015		MOV	R4,R5	!RESERVED WORD
2098				CLR	(R5)	!PRESET DATA=0
2099	011262	000277		SCC		
2100	011264	000244		CLZ		
2101	011266	005725		TST	(R5)+	!(R5)=000000, CC=0100
2102	011270	103402		BCS	TST2	
2103	011272	102401		BVS	TST2	
2104	011274	001401		BEQ	,+4	
2105	011276	104400		TST21	HLT	!ERROR! INCORRECT CC'S AS SHOWN ABOVE
2106						
2107	011300	005145		COM	=(R5)	!(R5)=177777, CC=1001
2108	011302	103001		BCC	COM4	
2109	011304	100401		BMI	,+4	
2110	011306	104400		COM41	HLT	!ERROR! INCORRECT CC'S AS SHOWN ABOVE
2111						
2112	011310	000241		CLC		
2113	011312	006024		ROR	(R4)+	!(R4)=077777, CC=0011
2114	011314	103002		BCC	ROR2	
2115	011316	102001		BVC	ROR2	
2116	011320	100001		BPL	,+4	
2117	011322	104400		ROR21	HLT	!ERROR! INCORRECT CC'S AS SHOWN ABOVE
2118						
2119	011324	000257		CCC		
2120	011326	005244		INC	=(R4)	!(R4)=100000, CC=1010
2121	011330	102002		BVC	INC4	
2122	011332	001401		BEQ	INC4	
2123	011334	100401		BMI	,+4	
2124	011336	104400		INC41	HLT	!ERROR! INCORRECT CC'S AS SHOWN ABOVE
2125						
2126	011340	000261		SEC		

2127	011342	000324		SWAB	(R4)+	!(R4)=000200, CC=1000
2128	011344	103401		BCS	SWAB2	
2129	011346	100401		BMI	,+4	
2130	011350	104400		SWAB21	HLT	!ERROR! INCORRECT CC'S AS SHOWN ABOVE
2131						
2132	011352	005425		NEG	(R5)+	!(R5)=177600, CC=1001
2133	011354	103001		BCC	NEG2	
2134	011356	100401		BMI	,+4	
2135	011360	104400		NEG21	HLT	!ERROR! INCORRECT CC'S AS SHOWN ABOVE
2136						
2137	011362	005044		CLR	=(R4)	!(R4)=000000, CC=0100
2138	011364	001401		BEQ	,+4	
2139	011366	104400		HLT		
2140						
2141	011370	000261		SEC		
2142	011372	006045		ROR	=(R5)	!(R5)=100000, CC=1010
2143	011374	000261		SEC		
2144	011376	005525		ADC	(R5)+	!(R5)=100001, CC=1000
2145	011400	102401		BVS	ADC2	
2146	011402	100401		BMI	,+4	
2147	011404	104400		ADC21	HLT	!ERROR! INCORRECT CC'S AS SHOWN ABOVE
2148						
2149	011406	000262		SEV		
2150	011410	006224		ASR	(R4)+	!(R4)=140000, CC=1001
2151	011412	103002		BCC	ASR2	
2152	011414	102401		BVS	ASR2	
2153	011416	100401		BMI	,+4	
2154	011420	104400		ASR21	HLT	!ERROR! INCORRECT CC'S AS SHOWN ABOVE
2155						
2156	011422	000262		SEV		
2157	011424	006144		ROL	=(R4)	!(R4)=100001, CC=1001
2158	011426	103002		BCC	ROL4	
2159	011430	102401		BVS	ROL4	
2160	011432	100401		BMI	,+4	
2161	011434	104400		ROL41	HLT	!ERROR! INCORRECT CC'S AS SHOWN ABOVE
2162						
2163	011436	005645		SBC	=(R5)	!(R5)=100000, CC=1000
2164	011440	103001		BCC	,+4	
2165	011442	104400		HLT		!ERROR! 'C' BIT FAILED TO CLEAR
2166						
2167	011444	005325		DEC	(R5)+	!(R5)=077777, CC=0010
2168	011446	103402		BCS	DEC2	
2169	011450	102001		BVC	DEC2	
2170	011452	100001		BPL	,+4	
2171	011454	104400		DEC21	HLT	!ERROR! INCORRECT CC'S AS SHOWN ABOVE
2172						
2173	011456	006324		ASL	(R4)+	!(R4)=177776, CC=1010
2174	011460	102401		BVS	,+4	
2175	011462	104400		HLT		
2176	011464	006344		ASL	=(R4)	!(R4)=177774, CC=1001
2177	011466	103003		BCC	ASL4	
2178	011470	102402		BVS	ASL4	
2179	011472	001401		BEQ	ASL4	
2180	011474	100401		BMI	,+4	

2181	011476	104400	ASL4:	HLT		;	ERROR	INCORRECT	CC'S	AS	SHOWN	ABOVE
2182												
2183	011500	022724		CMP	#177774,(R4)+							
2184	011504	001401		BEQ	,+4							
2185	011506	104400		HLT								
2186	011510	020405		CMP	R4,R5							
2187	011512	001401		BEQ	,+4							
2188	011514	104400		HLT								
2189	011516	104000		SCOPE								
2190												
2191												
2192	011520	000401		OR	,+4							
2193	011522	000000		WORD	0							
2194	011524	010705		MOV	PC,R5							
2195	011526	162705		SUB	#4,R5							
2196	011532	010500		MOV	R5,R0							
2197	011534	010002		MOV	R0,R2							
2198	011536	005202		INC	R2							
2199	011540	005010		CLR	(R0)							
2200												
2201	011542	000277		SCC								
2202	011544	000241		CLC								
2203	011546	105125		COMB	(R5)+							
2204	011550	103002		BCC	COMB2							
2205	011552	102401		BVS	COMB2							
2206	011554	100401		BMI	,+4							
2207	011556	104400	COMB2:	HLT								
2208												
2209	011560	105542		ADCB	=(R2)							
2210	011562	001401		BEQ	,+4							
2211	011564	104400		HLT								
2212	011566	105525		ADCB	(R5)+							
2213	011570	103401		BCC	ADCB2							
2214	011572	001001		BNE	,+4							
2215	011574	104400	ADCB2:	HLT								
2216												
2217	011576	000263		+SECI:SEV								
2218	011600	106045		RORB	=(R5)							
2219	011602	103003		BCC	RORB4							
2220	011604	102402		BVS	RORB4							
2221	011606	001401		BEQ	R0FB4							
2222	011610	100401		BMI	,+4							
2223	011612	104400	RORB4:	HLT								
2224												
2225	011614	000277		SCC								
2226	011616	106122		ROLB	(R2)+							
2227	011620	103403		BCC	ROLB2							
2228	011622	102402		BVS	ROLB2							
2229	011624	001401		BEQ	ROLB2							
2230	011626	100001		BPL	,+4							
2231	011630	104400	ROLB2:	HLT								
2232												
2233	011632	000257		CCG								
2234	011634	106225		ASRB	(R5)+							

2235	011636	103402		BCC	ASRB2							
2236	011640	102001		BVC	ASRB2							
2237	011642	100401		BMI	,+4							
2238	011644	104400	ASRB2:	HLT								
2239												
2240	011646	105242		NCB	=(R2)							
2241	011650	000277		SCC								
2242	011652	106222		ASRB	(R2)+							
2243	011654	103402		BCC	ASRB2A							
2244	011656	102401		BVS	ASRB2A							
2245	011660	100001		BPL	,+4							
2246	011662	104400	ASRB2A:	HLT								
2247												
2248	011664	000266		+SEI:SEV								
2249	011666	106345		ASLB	=(R5)							
2250	011670	103003		BCC	ASLB4							
2251	011672	102402		BVS	ASLB4							
2252	011674	001401		BEQ	ASLB4							
2253	011676	100401		BMI	,+4							
2254	011700	104400	ASLB4:	HLT								
2255												
2256	011702	105322		DECB	(R2)+							
2257	011704	103002		BCC	DECB2							
2258	011706	102001		BVC	DECB2							
2259	011710	100001		BPL	,+4							
2260	011712	104400	DECB2:	HLT								
2261												
2262	011714	105645		SBCB	=(R5)							
2263	011716	103402		BCC	SBCB4							
2264	011720	102401		BVS	SBCB4							
2265	011722	001401		BEQ	,+4							
2266	011724	104400	SBCB4:	HLT								
2267												
2268	011726	105442		NECB	=(R2)							
2269	011730	103002		BCC	NECB4							
2270	011732	102401		BVS	NECB4							
2271	011734	100401		BMI	,+4							
2272	011736	104400	NECB4:	HLT								
2273												
2274	011740	105725		TSTB	(R5)+							
2275	011742	103401		BCC	TSTB2							
2276	011744	001401		BEQ	,+4							
2277	011746	104400	TSTB2:	HLT								
2278												
2279	011750	105722		TSTB	(R2)+							
2280	011752	001401		BEQ	TSTB2A							
2281	011754	100401		BMI	,+4							
2282	011756	104400	TSTB2A:	HLT								
2283												
2284	011760	000261		SEC								
2285	011762	000342		SWAB	=(R2)							
2286	011764	103401		BCC	SWAB4							
2287	011766	100401		BMI	,+4							
2288	011770	104400	SWAB4:	HLT								

2289									
2290	011772	000277		SCC					
2291	011774	105225		INCB	(R5)+				
2292	011776	103003		RCC	INCB2				(R0)=000001*(0004)(201),CC=0000
2293	012000	102402		BVS	INCR2				
2294	012002	001401		BEQ	INCB2				
2295	012004	100001		BPL	,+4				
2296	012006	104400		INCB2:	HLT				
2297									
2298	012010	022227	000601	CHP	(R2)+,#000601				ICHECK END RESULT
2299	012014	001401		BEQ	,+4				
2300	012016	104400		HLT					
2301	012020	020205		CHP	R2,R5				ICHECK REGISTERS
2302	012022	001401		BEQ	,+4				
2303	012024	104400		HLT					
2304	012026	104000		SCOPE					
2305									
2306				ICHECK	UNARY WORD OPS USING ADDRESS MODES 3 AND 5				
2307	012030	000402		BR	,+6				RESERVE 2 WORDS
2308	012032	000000		WORD	0				FOR THE ADDRESS
2309	012034	000000		WORD	0				AND 1 FOR DATA
2310	012036	010703		MOV	PC,R3				
2311	012040	102700	000004	SUB	#4,R3				
2312	012044	005010		CLR	(R3)				PRESET DATA
2313	012046	103000		MOV	R3,R0				R0 POINTS TO DATA WORD
2314	012050	005743		TST	-(R3)				
2315	012052	010013		MOV	R0,(R3)				
2316	012054	103004		MOV	R3,R4				
2317									
2318	012056	000297		CCC					
2319	012060	005733		TST	*(R3)+				(R0)=000000,CC=0100
2320	012062	001401		BEQ	,+4				
2321	012064	104400		HLT					
2322									
2323	012066	000261		SEC					
2324	012070	006093		ROR	*(R3)				(R0)=100000,CC=1010
2325	012072	103400		BCS	ROR5				
2326	012074	102001		BVC	ROR5				
2327	012076	100401		BMI	,+4				
2328	012100	104400		ROR5:	HLT				
2329									
2330	012102	000297		CCC					
2331	012104	006234		ASR	*(R0)+				(R0)=140000,CC=1010
2332	012106	102001		BVC	ASR0				
2333	012110	100401		BMI	,+4				
2334	012112	104400		ASR3:	HLT				
2335									
2336	012114	000250		CLN					
2337	012116	006333		ASL	*(R3)+				(R0)=100000,CC=1001
2338	012120	103002		BCC	ASL3				
2339	012122	102401		BVS	ASL3				
2340	012124	100401		BMI	,+4				
2341	012126	104400		ASL3:	HLT				
2342									

2343	012130	000277		SCC					
2344	012132	005394		DEC	*(R4)				(R0)=077777, CC=0010
2345	012134	103003		BCC	DEC5				
2346	012136	102002		BVC	DEC5				
2347	012140	001401		BEQ	DEC5				
2348	012142	100001		BPL	,+4				
2349	012144	104400		DEC5:	HLT				
2350									
2351	012146	005453		NEG	*(R3)				(R0)=100001, CC=1001
2352	012150	103002		BCC	NEG5				
2353	012152	102401		BVS	NEG5				
2354	012154	100401		BMI	,+4				
2355	012156	104400		NEG5:	HLT				
2356									
2357	012160	000262		SEV					
2358	012162	005134		COM	*(R6)+				(R0)=077776, CC=0001
2359	012164	103001		BCC	COM3				
2360	012166	102001		BVC	,+4				
2361	012170	104400		COM3:	HLT				
2362									
2363	012172	005233		INC	*(R3)+				(R0)=077777, CC=0001
2364	012174	103001		BCC	INC3				
2365	012176	100001		BPL	,+4				
2366	012200	104400		INC3:	HLT				
2367									
2368	012202	005594		ADC	*(R4)				(R0)=100000, CC=1010
2369	012204	103402		BCS	ADC5				
2370	012206	102001		BVC	ADC5				
2371	012210	100401		BMI	,+4				
2372	012212	104400		ADC5:	HLT				
2373									
2374	012214	000297		CCC					
2375	012216	006134		ROL	*(R4)+				(R0)=000000,CC=0111
2376	012220	103002		BCC	ROL3				
2377	012222	102001		BVC	ROL3				
2378	012224	001401		BEQ	,+4				
2379	012226	104400		ROL3:	HLT				
2380									
2381	012230	005253		INC	*(R3)				(R0)=000001, CC=0001
2382	012232	005694		SBC	*(R4)				(R0)=000000, CC=0100
2383	012234	103401		BCS	SBC5				
2384	012236	001401		BEQ	,+4				
2385	012240	104400		SBC5:	HLT				
2386	012242	104000		SCOPE					
2387									
2388				ICHECK	UNARY BYTE OPS USING ADDRESS MODES 3 AND 5				
2389	012244	000403		BR	,+10				RESERVE 3 WORDS
2390	012246	000000		WORD	0				FOR EVEN BYTE ADDRESS
2391	012250	000000		WORD	0				FOR ODD BYTE ADDRESS
2392	012252	000000		WORD	0				AND 1 FOR DATA
2393	012254	010702		MOV	PC,R2				
2394	012256	025742		TST	-(R2)				BACK R2 UP TO
2395	012260	025742		TST	-(R2)				DATA WORD
2396	012262	010200		MOV	R2,R0				R0 POINTS TO THE DATA WORD

2397	012264	005010	CLR	(R0)	;	PRESET DATA
2398	012266	005742	TST	*(R2)	;	BACK R2 UP TO
2399	012270	005742	TST	*(R2)	;	EVEN BYTE ADDRESS WORD
2400	012272	010022	MOV	R0,(R2)+	;	LOAD SOL CLASS
2401	012274	005200	INC	R0	;	ODD BYTE ADDRESS
2402	012276	010022	MOV	R0,(R2)+	;	LOAD ODD BYTE ADDRESS
2403	012300	210200	MOV	R2,R0	;	RESET R0
2404	012302	010200	MOV	R2,R5		
2405						
2406	012304	105152	COMB	*(R2)	;	(R0)=177400,CC=1001
2407	012306	103001	BCC	COMB5		
2408	012310	100401	BMI	,+4		
2409	012312	104400	COMB5;	HLT		
2410						
2411	012314	105752	TSTB	*(R2)	;	(R0)=177400, CC=0100
2412	012316	001401	REQ	,+4		
2413	012320	104400	HLT			
2414						
2415	012322	000262	SEV			
2416	012324	106255	ASRB	*(R5)	;	(R0)=177400, CC=1001
2417	012326	103002	BCC	ASRB5		
2418	012330	102401	BVS	ASRB5		
2419	012332	100401	BMI	,+4		
2420	012334	104400	ASRB5;	HLT		
2421						
2422	012336	105232	INCB	*(R2)+	;	(R0)=177401, CC=0000
2423	012340	103001	BCC	INCB3		
2424	012342	100001	BPL	,+4		
2425	012344	104400	INCB3;	HLT		
2426						
2427	012346	000241	CLC			
2428	012350	106055	RORB	*(R5)	;	(R0)=177400, CC=0111
2429	012352	103003	RCC	RORB5		
2430	012354	102002	RVC	RORB5		
2431	012356	001001	BNE	RORB5		
2432	012360	100001	BPL	,+4		
2433	012362	104400	RORB5;	HLT		
2434						
2435	012364	106332	ASLB	*(R2)+	;	(R0)=177000, CC=1001
2436	012366	103002	BCC	ASLB3		
2437	012370	102401	BVS	ASLB3		
2438	012372	100401	BMI	,+4		
2439	012374	104400	ASLB3;	HLT		
2440						
2441	012376	105552	ADCB	*(R2)	;	(R0)=177400, CC=1000
2442	012400	103401	BCC	ADCB5		
2443	012402	100401	BMI	,+4		
2444	012404	104400	ADCB5;	HLT		
2445						
2446	012406	000277	SCC			
2447	012410	106135	ROLB	*(R5)+	;	(R0)=177401, CC=0000
2448	012412	101402	BLOS	ROLB3	;	BRANCH IF C OR Z IS SET
2449	012414	102401	BVS	ROLB3		
2450	012416	100001	BPL	,+4		

2451	012420	104400	ROLB3;	HLT		
2452						
2453	012422	000352	SWAB	*(R2)	;	(R0)=000777, CC=1000
2454	012424	100401	BMI	,+4		
2455	012426	104400	HLT			
2456						
2457	012430	000261	SEC			
2458	012432	105635	SBCB	*(R5)+	;	(R0)=000377, CC=0100
2459	012434	103401	BCC	SBCB3		
2460	012436	001401	BEO	,+4		
2461	012440	104400	SBCB3;	HLT		
2462						
2463	012442	105432	NECB	*(R2)+	;	(R0)=000001
2464	012444	105352	DECB	*(R2)	;	(R0)=000000, CC=0101
2465	012446	103001	BCC	DECB5		
2466	012450	001401	REQ	,+4		
2467	012452	104400	DECB5;	HLT		
2468	012454	104000	SCOPE			
2469						
2470						
2471	012456	005027	;	CHECK UNARY WORD OPS USING ADDRESS MODE 6 (PC)		
2472	012460	000000	UNM6;	CLR (PC)+	;	PRESET DATA # 0
2473	012462	010700	,WORD	0	;	RESERVED FOR DATA
2474	012464	024040	MOV	PC,RB		
			CMF	=(RB),-(R0)	;	R0 POINTS TO DATA WORD

2475	012466	002277		SCC		
2476	012470	006167	177764	ROL	UWM6	;(R0)=000001,CC=0000
2477	012474	103403		BVS	ROL6	
2478	012476	102402		RVS	ROL6	
2479	012500	001401		REQ	ROL^	
2480	012502	100001		BPL	,+4	
2481	012504	104400		ROL61	HLT	
2482						
2483	012506	005167	177746	COM	UWM6	;(R0)=177776, CC=1001
2484	012512	103002		BCC	COM6	
2485	012514	102401		BVS	COM6	
2486	012516	100401		BMI	,+4	
2487	012520	104400		COM61	HLT	
2488	012522	006267	177732	ASR	UWM6	;(R0)=177777, CC=1010
2489	012526	103402		BVS	ASR6	
2490	012530	102001		BVC	ASR6	
2491	012532	100401		BMI	,+4	
2492	012534	104400		ASR61	HLT	
2493						
2494	012536	000277		SCC		
2495	012540	005467	177714	NEB	UWM6	;(R0)=000001, CC=0001
2496	012544	103003		BCC	NEG6	
2497	012546	102402		BVS	NEG6	
2498	012550	001401		REQ	NEG6	
2499	012552	100001		BPL	,+4	
2500	012554	104400		NEG61	HLT	
2501						
2502	012556	000277		SCC		
2503	012560	005067	177674	ROR	UWM6	;(R0)=100000, CC=1001
2504	012564	103003		BCC	ROR6	
2505	012566	102402		BVS	ROR6	
2506	012570	001401		REQ	ROR6	
2507	012572	100401		BMI	,+4	
2508	012574	104400		ROR61	HLT	
2509						
2510	012576	005667	177656	SBC	UWM6	;(R0)=077777, CC=0010
2511	012602	103402		BVS	SBC6	
2512	012604	102001		BVC	SBC6	
2513	012606	100001		BPL	,+4	
2514	012610	104400		SBC61	HLT	
2515						
2516	012612	000242		CLV		
2517	012614	005267	177640	INC	UWM6	;(R0)=100000, CC=1011
2518	012620	103403		BCC	INC6	
2519	012622	102002		BVC	INC6	
2520	012624	001401		REQ	INC6	
2521	012626	100401		BMI	,+4	
2522	012630	104400		INC61	HLT	
2523						
2524	012632	006267	177622	ASR	UWM6	;(R0)=140000, CC=1010
2525	012636	000241		SEC		
2526	012640	005367	177614	ASL	UWM6	;(R0)=100000, CC=1001
2527	012644	103002		BCC	ASL6	
2528	012646	102401		BVS	ASL6	

2529	012650	100401		BMI	,+4	
2530	012652	104400		ASL61	HLT	
2531						
2532	012654	005367	177600	DEC	UWM6	;(R0)=077777, CC=0011
2533	012660	103002		BCC	DEC6	
2534	012662	102001		BVC	DEC6	
2535	012664	100001		BPL	,+4	
2536	012666	104400		DEC61	HLT	
2537						
2538	012670	005567	177564	ADC	UWM6	;(R0)=100000, CC=1010
2539	012674	103402		BVS	ADC6	
2540	012676	102001		BVC	ADC6	
2541	012700	100401		BMI	,+4	
2542	012702	104400		ADC61	HLT	
2543	012704	000242		CLV		
2544	012706	000367	177546	SWAB	UWM6	
2545	012712	100401		BMI	,+4	
2546	012714	104400		HLT		
2547	012716	002710	000200	CHP	#200,(R0)	
2548	012722	001401		REQ	,+4	
2549	012724	104400		HLT		
2550	012726	104000		SCOPE		
2551						
2552						
2553	012730	012700	013272	ICHECK	UNARY BYTE OPS (EVEN/ODD) USING ADDRESS MODE 6 (PC)	
2554	012734	003700	001004	MOV	#UBM6,R0	
2555	012740	005067	000326	ADD	#FACTOR,R0	IR0 POINTS TO ADDRESS OF DATA
2556	012744	000277		CLR	UBM6	ICLEAR DATA
2557	012746	000244		SCC		
2558	012750	105767	000316	CLE		
2559	012754	103403		TSTB	UBM6	
2560	012756	102402		BVS	TSTB6	
2561	012760	001001		BNE	TSTB6	
2562	012762	100001		BPL	,+4	
2563	012764	104400		TSTB61	HLT	
2564						
2565	012766	000257		CCC		
2566	012770	105767	000277	TSTB	UBM6+1	;TEST ODD BYTE
2567	012774	001401		REQ	,+4	
2568	012776	104400		HLT		
2569						
2570	013000	105667	000266	SBCB	UBM6	;(R0)=000000, CC=0100
2571	013004	103402		BVS	SBCB6	
2572	013006	102401		BVS	SBCB6	
2573	013010	001401		REQ	,+4	
2574	013012	104400		SBCB61	HLT	
2575						
2576	013014	000241		1\$	SEC	
2577	013016	105267	000250	INCB	UBM6	;LOOP UNTIL (R0)=077600, CC=1011
2578	013022	100403		BMI	2\$	
2579	013024	105567	000243	ADCB	UBM6+1	;INCB INST INCREMENTS EVEN BYTE
2580	013030	000771		BR	1\$;ADCB INCREMENTS ODD BYTE
2581	013032	103001		RCC	INCB6	
2582	013034	102401		RVS	,+4	

2583	013036	104400		INCB6I	HLT		
2584							
2585	013040	106367	000226	ASLB	UBM6	;(R0)=077400, CC#0111	
2586	013044	103003		BCC	ASLB6		
2587	013046	102002		BVC	ASLB6		
2588	013050	001001		BNE	ASLB6		
2589	013052	100001		BPL	,+4		
2590	013054	104400		ASLB6I	HLT		
2591							
2592	013056	000242		CLV			
2593	013060	105567	000207	ADCB	UBM6+1	;(R0)=100000, CC#1010	
2594	013064	103402		BCC	ADCB6		
2595	013066	102001		BVC	ADCB6		
2596	013070	100401		BMI	,+4		
2597	013072	104400		ADCB6I	HLT		
2598							
2599	013074	000241		SEC			
2600	013076	106067	000171	RORB	UBM6+1	;(R0)=140000, CC#1010	
2601	013102	103402		BCC	RORB6		
2602	013104	102001		BVC	RORB6		
2603	013106	100401		BMI	,+4		
2604	013110	104400		RORB6I	HLT		
2605							
2606	013112	105167	000154	COMB	UBM6	;(R0)=140377 CC#1001	
2607	013116	103002		BCC	COMB6		
2608	013120	102401		BVS	COMB6		
2609	013122	100401		BMI	,+4		
2610	013124	104400		COMB6I	HLT		
2611							
2612	013126	000242		SEV			
2613	013130	105467	000137	NEGB	UBM6+1	;(R0)=040377, CC#0001	
2614	013134	103002		BCC	NEGB6		
2615	013136	102401		BVS	NEGB6		
2616	013140	100001		BPL	,+4		
2617	013142	104400		NEGB6I	HLT		
2618							
2619	013144	106167	000123	ROLB	UBM6+1	;(R0)=100777, CC#1010	
2620	013150	103402		BCC	ROLB6		
2621	013152	102001		BVC	ROLB6		
2622	013154	100401		BMI	,+4		
2623	013156	104400		ROLB6I	HLT		
2624							
2625	013160	106267	000106	ASRB	UBM6	;(R0)=100777, CC#1001	
2626	013164	103002		BCC	ASRB6		
2627	013166	102401		BVS	ASRB6		
2628	013170	100401		BMI	,+4		
2629	013172	104400		ASRB6I	HLT		
2630							
2631	013174	105267	000072	INCB	UBM6	;(R0)=100400, CC#0101	
2632	013200	103002		BCC	INCB6A		
2633	013202	102401		BVS	INCB6A		
2634	013204	001401		BEQ	,+4		
2635	013206	104400		INCB6AI	HLT		
2636							

2637	013210	105367	000057	DECB	UBM6+1	;(R0)=100000, CC#1001	
2638	013214	103003		BCC	DECB6A		
2639	013216	102402		BVS	DECB6A		
2640	013220	001401		BEQ	DECB6A		
2641	013222	100401		BMI	,+4		
2642	013224	104400		DECB6AI	HLT		
2643							
2644	013226	000367	000040	SWAB	UBM6	;(R0)=000200, CC#1000	
2645	013232	103401		BCC	SWAB6		
2646	013234	100401		BMI	,+4		
2647	013236	104400		SWAB6I	HLT		
2648							
2649	013240	106167	000026	ROLB	UBM6	;(R0)=000000, CC#0111	
2650	013244	103002		BCC	ROLB6A		
2651	013246	102001		BVC	ROLB6A		
2652	013250	001401		BEQ	,+4		
2653	013252	104400		ROLB6AI	HLT		
2654							
2655	013254	005767	000012	TST	UBM6	;(R0)=000000, CC#0100	
2656	013260	103402		BCC	TST6		
2657	013262	102401		BVS	TST6		
2658	013264	001401		BEQ	,+4		
2659	013266	104400		TST6I	HLT		
2660							
2661	013270	000401		BR	,+4	;(RESERVE A WORD	
2662	013272	000000		UBM6I	,WORD 0	;(WORD RESERVED FOR DATA	
2663	013274	104000		SCOPE			
2664	013276	010702		MOV	PC,R2		
2665	013300	062702	000012	ADD	#12,R2		
2666	013304	012707	001152	MOV	#RELOC,PC	;(GO RELOCATE PROGRAM CODE	
2667	013310	000000		REL11I	,WORD 0		
2668							
2669							
2670							
2671							
2672							
2673	013312	010700		REL2I	MOV PC,R0	;(GET PC	
2674	013314	005740					
2675	013316	010037	001010	TST	-(R0)	;(R0 CONTAINS THE ADDRESS OF REL2	
2676	013322	012737	000002 005176	MOV	R0,#FRSTAD	;(SAVE	
2677	013330	004737	005166	MOV	#2,#SECT	;(SET SECTION #	
2678	013334	013767	005172 003744	JSR	PC,#LDDISP	;(LOAD DISPLAY GEG	
2679	013342	010700		MOV	#DISPLY,REL22		
2680	013344	102700	213344	MOV	PC,R0	;(GET CURRENT PC	
2681	013350	010037	001024	SUB	#,R0	;(SUBTRACT RELOCATION FACTOR	
2682	013354	010701		MOV	R0,#FACTOR	;(SAVE RELOCATION FACTOR	
2683				MOV	PC,R1	;(SET NEW SCOPE PTR	
2684							
2685	013356	002403		UBM7I	BR UW7	;(RESERVE 3 WORDS FOR ADDRESSES & DATA	
2686	013360	000000			,WORD 0	;(CONTAINS ADDRESS OF UW7	
2687	013362	000000			,WORD 0	;(CONTAINS DATA	
2688	013364	000000			,WORD 0	;(CONTAINS ADDRESS OF UW7	
2689							
2690	013366	010700		UW7I	MOV PC,R0		

2691	013370	005740		TST	=(R0)	
2692	013372	005740		TST	=(R0)	
2693	013374	005040		CLR	=(R0)	ICLEAR TEST DATA
2694	013376	010002		MOV	R0,R2	
2695	013400	010240		MOV	R2,=(R0)	!SET UP ADDRESS
2696	013402	005720		TST	(R0)+	!MOVE R0 TO NEXT ADDRESS
2697	013404	005720		TST	(R0)+	
2698	013406	010210		MOV	R2,(R0)	!SET NEXT ADDRESS
2699	013410	010200		MOV	R2,R0	!SET R0 POINTING TO DATA
2700	013412	000277		SCC		
2701	013414	000244		CLZ		
2702	013416	005772	000002	TST	#2(2)	!(R0)=000000, CC=0100
2703	013422	001401		BEQ	,+4	
2704	013424	104400		HLT		
2705						
2706	013426	000277		SCC		
2707	013430	005672	177776	SBC	#-2(2)	!(R0)=177777, CC=1001
2708	013434	103002		BCC	SBC#	
2709	013436	102401		BVS	SBC#	
2710	013440	100401		BMI	,+4	
2711	013442	104400		HLT		
2712						
2713	013444	000277		SCC		
2714	013446	000241		CLC		
2715	013450	006372	000002	ASL	#2(2)	!(R0)=177776, CC=1001
2716	013454	103002		BCC	ASL#	
2717	013456	102401		BVS	ASL#	
2718	013460	100401		BMI	,+4	
2719	013462	104400		HLT		
2720						
2721	013464	000257		CCC		
2722	013466	005372	000002	DEC	#2(2)	!(R0)=177775, CC=1000
2723	013472	103402		BCC	DEC#	
2724	013474	102401		BVS	DEC#	
2725	013476	100401		BMI	,+4	
2726	013500	104400		HLT		
2727						
2728	013502	000262		SEV		
2729	013504	006272	177776	ASR	#-2(2)	!(R0)=177776, CC=1001
2730	013510	103002		BCC	ASR#	
2731	013512	102401		BVS	ASR#	
2732	013514	100401		BMI	,+4	
2733	013516	104400		HLT		
2734						
2735	013520	000241		CLC		
2736	013522	000262		SEV		
2737	013524	006072	177776	WOR	#-2(2)	!(R0)=077777, CC=0000
2738	013530	101402		BLOS	ROR#	!BRANCH IF C OR Z IS SET
2739	013532	102401		BVS	ROR#	
2740	013534	100001		BPL	,+4	
2741	013536	104400		HLT		
2742						
2743	013540	000262		SEV		
2744	013542	005472	000002	NEG	#2(2)	!(R0)=100001, CC=1001

2745	013546	103002		BCC	NEG#	
2746	013550	102401		BVS	NEG#	
2747	013552	100401		BMI	,+4	
2748	013554	104400		HLT		
2749						
2750	013556	000250		CLN		
2751	013560	000372	177776	SWAB	#-2(2)	!(R0)=000600, CC=1000
2752	013564	103401		BCC	SWAB#	
2753	013566	100401		BMI	,+4	
2754	013570	104400		HLT		
2755						
2756	013572	000262		SEV		
2757	013574	005172	000002	COM	#2(2)	!(R0)=177177, CC=1001
2758	013600	103002		BCC	COM#	
2759	013602	102401		BVS	COM#	
2760	013604	100401		BMI	,+4	
2761	013606	104400		HLT		
2762						
2763	013610	000372	000002	SWAB	#2(2)	!(R0)=077776, CC=1000
2764	013614	100401		BMI	,+4	
2765	013616	104400		HLT		
2766						
2767	013620	000277		SCC		
2768	013622	005572	177776	ADC	#-2(2)	!(R0)=077777, CC=0000
2769	013626	103402		BCC	ADC#	
2770	013630	102401		BVS	ADC#	
2771	013632	100001		BPL	,+4	
2772	013634	104400		HLT		
2773						
2774	013636	005272	000002	INC	#2(2)	!(R0)=100000, CC=1010
2775	013642	102001		BVC	INC#	
2776	013644	100401		BMI	,+4	
2777	013646	104400		HLT		
2778						
2779	013650	000257		CCC		
2780	013652	006172	177776	ROL	#-2(2)	!(R0)=000000, CC=0111
2781	013656	103002		BCC	ROL#	
2782	013660	102001		BVC	ROL#	
2783	013662	001401		BEQ	,+4	
2784	013664	104400		HLT		
2785	013666	104000		SCOPE		
2786						
2787						
2788	013670	005720		ICHECK	UNARY BYTE OPS USING ADDRESS MODE 7	
2789	013672	005210		TST	(R0)+	
2790	013674	005740		INC	(R0)	!WORD FOLLOWING UWM7 CONTAINS ADDRESS
2791	013676	005010		TST	-(R0)	!IF ODD BYTE, R0 POINTS TO DATA WORD
2792	013700	010701		CLR	(R0)	!PRESET DATA
2793				MOV	PC,R1	!SET SCOPE PTR
2794				INOTE	#2(2)	!REFERENCES THE ODD BYTE, AND #-2(2) REFERENCES THE EVEN BYTE.
2795	013702	000263		+SECTISEV		!SET C AND V
2796	013704	105672	000002	SBCB	#2(2)	!(R0)=177400, CC=1001
2797	013710	103003		BCC	SBCB#	
2798	013712	102402		BVS	SBCB#	

2799	013714	001401		REQ	SBCB7		
2800	013716	100401		BMI	,+4		
2801	013722	104400		SBCB7:	HLT		
2802							
2803	013722	000277		SCC			ISL CONDITION CODES
2804	013724	105572	177776	ADCB	0-2(2)		;(R0)=177401, CC=0000
2805	013730	103403		RCS	ADCB7		
2806	013732	102402		CVS	ADCB7		
2807	013734	001401		REQ	ADCB7		
2808	013736	100001		RPL	,+4		
2809	013740	104400		ADCB7:	HLT		
2810							
2811	013742	105172	177776	COMB	0-2(2)		;(R0)=177776, CC=1001
2812	013746	103002		BCC	COMB7		
2813	013750	102401		BVS	COMB7		
2814	013752	100401		BMI	,+4		
2815	013754	104400		COMB7:	HLT		
2816							
2817	013756	000241		CLC			ICLEAR CARRY
2818	013760	106072	000002	RORB	0-2(2)		;(R0)=077776, CC=0011
2819	013764	103002		BCC	RORB7		
2820	013766	102001		BVC	RORB7		
2821	013770	100001		RPL	,+4		
2822	013772	104400		RORB7:	HLT		
2823							
2824	013774	105272	000002	INCB	0-2(2)		;(R0)=100376, CC=1011
2825	014000	103002		BCC	INCB7		
2826	014002	102001		BVC	INCB7		
2827	014004	100401		BMI	,+4		
2828	014006	104400		INCB7:	HLT		
2829							
2830	014010	105372	177776	DECB	0-2(2)		;(R0)=100375, CC=1001
2831	014014	103002		BCC	DECB7		
2832	014016	102401		BVS	DECB7		
2833	014020	100401		BMI	,+4		
2834	014022	104400		DECB7:	HLT		
2835							
2836	014024	106372	000002	ASLB	0-2(2)		;(R0)=000375, CC=0111
2837	014030	103002		BCC	ASLB7		
2838	014032	102001		BVC	ASLB7		
2839	014034	001401		REQ	,+4		
2840	014036	104400		ASLB7:	HLT		
2841							
2842	014040	000241		CLC			ICLEAR CARRY
2843	014042	106272	177776	ASRB	0-2(2)		;(R0)=000376, CC=1001
2844	014046	103002		BCC	ASRB7		
2845	014050	102401		BVS	ASRB7		
2846	014052	100401		BMI	,+4		
2847	014054	104400		ASRB7:	HLT		
2848							
2849	014056	105472	000002	NEGB	0-2(2)		;(R0)=000376, CC=0100
2850	014062	103402		BCC	NEGB7		
2851	014064	102401		BVS	NEGB7		
2852	014066	001401		REQ	,+4		

2853	014070	104400		NEGB7:	HLT		
2854							
2855	014072	000262		SEV			
2856	014074	106172	177776	ROLB	0-2(2)		;(R0)=000374, CC=1001
2857	014100	103002		BCC	ROLB7		
2858	014102	102401		BVS	ROLB7		
2859	014104	100401		BMI	,+4		
2860	014106	104400		ROLB7:	HLT		
2861							
2862	014110	105272	177776	INCB	0-2(2)		;(R0)=000375, CC=1001
2863	014114	105272	177776	INCB	0-2(2)		;(R0)=000376, CC=1001
2864	014120	105572	177776	ADCB	0-2(2)		;(R0)=000377, CC=1000
2865	014124	105172	177776	COMB	0-2(2)		;(R0)=000000, CC=0100
2866	014130	001401		REQ	,+4		
2867	014132	104400		HLT			
2868	014134	104000		SCOPE			
2869							
2870							
2871	014136	000277		SCC			ICSET CONDITION CODES
2872	014140	010700		MOV	PC,R0		IR0=PC, CC=X001
2873	014142	103002		BCC	MOV0		
2874	014144	102401		BVS	MOV0		
2875	014146	001001		BNE	,+4		
2876	014150	104400		MOV0:	HLT		
2877							
2878	014152	010002		MOV	R0,R2		IR2=R0
2879	014154	000262		SEV			ICSET V
2880	014156	100002		SUB	R0,R2		IR2=000000, CC=0100
2881	014160	103402		SUB0			
2882	014162	102401		BCC	SUB0		
2883	014164	001401		BVS	SUB0		
2884	014166	104400		REQ	,+4		
2885				SUB0:	HLT		
2886							
2887	014170	000244		CLC			
2888	014172	010203		MOV	R2,R3		IR2=R3=000000, CC=0100
2889	014174	103401		BCC	MOV0A		
2890	014176	001401		REQ	,+4		
2891	014200	104400		MOV0A:	HLT		
2892							
2893	014202	000257		CDC			
2894	014204	000272		+SEV;SEN			ICSET V & N
2895	014206	020203		CMP	R2,R3		IR2=R3=000000, CC=0100
2896	014210	103403		BCC	CMP0		
2897	014212	102402		BVS	CMP0		
2898	014214	001001		BNE	CMP0		
2899	014216	100001		RPL	,+4		
2900	014220	104400		CMP0:	HLT		
2901							
2902	014222	010002		MOV	R0,R2		IR0=R2
2903	014224	010203		MOV	R2,R3		IR0=R2=R3
2904	014226	060203		ADD	R2,R3		IR3=2*R0
2905	014230	006302		ASL	R2		IR2=2*R0
2906	014232	020203		CMP	R2,R3		IR2=R3=2*R0
2907	014234	001401		REQ	,+4		

2907	014236	104400	HLT		ERROR! CHECK ADD INSTRUCTION
2908					
2909					
2910					
2911	014240	005002	CLR	R2	
2912	014242	005202	INC	R2	
2913	014244	000402	BR	2\$	
2914	014246	006302	ASL	R2	
2915	014250	100407	BMI	4\$	
2916	014252	010205	MOV	R2,R5	
2917	014254	000277	SCC		
2918	014256	030205	BIT	R2,R5	IR2=R5
2919	014260	103002	BCC	3\$	
2920	014262	102401	BVS	3\$	
2921	014264	001370	BNE	1\$	
2922	014266	104400	HLT		
2923	014270	010205	MOV	R2,R5	
2924	014272	000257	CCC		
2925	014274	030205	BIT	R2,R5	
2926	014276	100401	BMI	,+4	
2927	014300	104400	HLT		
2928					
2929	014302	005002	CLR	R2	
2930	014304	000277	SCC		
2931	014306	050002	BIS	R0,R2	
2932	014310	103002	BCC	BIS0	
2933	014312	102401	BVS	BIS0	
2934	014314	001001	BNE	,+4	
2935	014316	104400	HLT		
2936					
2937	014320	010003	MOV	R0,R3	
2938	014322	000277	SCC		
2939	014324	000244	CLZ		
2940	014326	040003	RIC	R0,R3	
2941	014330	103003	BCC	BIC0	
2942	014332	102402	BVS	BIC0	
2943	014334	001001	BNE	BIC0	
2944	014336	100001	BPL	,+4	
2945	014340	104400	HLT		
2946					
2947	014342	010004	MOV	R0,R4	
2948	014344	005104	COM	R4	
2949	014346	040004	BIC	R0,R4	
2950	014350	005104	COM	R4	
2951	014352	020004	CHP	R0,R4	
2952	014354	001401	BEQ	,+4	
2953	014356	104400	HLT		
2954					
2955	014360	010004	MOV	R0,R4	
2956	014362	005104	COM	R4	
2957	014364	010403	MOV	R4,R3	
2958	014366	050003	BIS	R0,R3	
2959	014370	103001	BCC	BIS0A	
2960	014372	100401	BMI	,+4	

2961	014374	104400	HLT		
2962	014376	005203	INC	R3	
2963	014400	001401	BEQ	,+4	
2964	014402	104400	HLT		
2965	014404	010304	MOV	R3,R4	IR3=R4=0
2966	014406	005103	COM	R3	IR3=17777
2967	014410	002601	SEC		IR3=C
2968	014412	006004	ROR	R4	IR4=100000
2969	014414	003304	ADD	R3,R4	IR3=17777,R4=07777,CC=0E11
2970	014416	103003	BCC	ADD0	
2971	014420	102002	BVC	ADD0	
2972	014422	001401	BEQ	ADD0	
2973	014424	100001	BPL	,+4	
2974	014426	104400	HLT		
2975	014430	010700	MOV	PC,R0	
2976	014432	022020	CHP	(R0),R0	
2977	014434	020007	CHP	R0,PC	
2978	014436	001401	BEQ	,+4	
2979	014440	104400	HLT		
2980					
2981	014442	010700	MOV	PC,R0	
2982	014444	062700	ADD	#10,R0	
2983	014450	010002	MOV	R0,R2	
2984	014452	020700	CHP	PC,R0	
2985	014454	001002	BNE	CHP0A	
2986	014456	020200	CHP	R2,R0	
2987	014460	001401	BEQ	,+4	
2988	014462	104400	HLT		
2989	014464	104000	SCOPE		
2990					
2991					
2992					
2993	014466	000402	BR	,+6	IR3=17777
2994	014470	000000	WORD	0	IR4=000001
2995	014472	000000	WORD	0	IR5=17777
2996	014474	010704	MOV	PC,R4	IR6=100000
2997	014476	005744	TST	=(R4)	IR7=17777,(R4)=000000,CC=0101
2998	014500	005044	CLR	=(R4)	
2999	014502	010403	MOV	R4,R3	
3000	014504	005043	CLR	=(R3)	
3001					
3002	014506	005113	COM	(R3)	IR3=17777
3003	014510	005214	INC	(R4)	IR4=000001
3004	014512	000262	SEV		IR5=17777,(R4)=000000,CC=0101
3005	014514	061314	ADD	(R3),(R4)	
3006	014516	103002	BCC	ADD1	
3007	014520	102401	BVS	ADD1	
3008	014522	001401	BEQ	,+4	
3009	014524	104400	HLT		
3010					
3011	014526	000277	SCC		
3012	014530	000250	CLN		
3013	014532	021314	CHP	(R3),(R4)	IR6=17777,(R4)=000000,CC=1000
3014	014534	103403	BCS	CHP1	

3015	014536	102402	BVS	CMP3		
3016	014540	001401	BEQ	CMP3		
3017	014542	100401	BMI	,+4		
3018	014544	104400	CMP1:	HLT		
3019						
3020	014546	000277	SCC			
3021	014550	000244	CLZ			
3022	014552	031314	BIT	(R3),(R4)		(R3)=177777,(R4)=000000,CC=0101
3023	014554	103002	BCC	BIT3		
3024	014556	102401	BVS	BIT3		
3025	014560	001401	BEQ	,+4		
3026	014562	104400	BIT1:	HLT		
3027						
3028	014564	000277	SCC			
3029	014566	000245	+CLC:CLZ			
3030	014570	005114	COM	(R4)		(R4)=177777
3031	014572	161314	SUB	(R3),(R4)		(R3)=177777,(R4)=000000,CC=0100
3032	014574	103402	BCC	SUB3		
3033	014576	102401	BVS	SUB3		
3034	014600	001401	BEQ	,+4		
3035	014602	104400	SUB1:	HLT		
3036						
3037	014604	105013	CLRB	(R3)		(R3)=177400
3038	014606	000313	SWAB	(R3)		(R3)=000377
3039	014610	000270	SEN			
3040	014612	011314	MOV	(R3),(R4)		(R3)=(R4)=000377
3041	014614	100001	BPL	,+4		
3042	014616	104400	HLT			
3043	014620	000314	SWAB	(R4)		(R3)=000377,(R4)=177400
3044	014622	000263	+SEC:SEV			SET C & V
3045	014624	051314	BIS	(R3),(R4)		(R3)=000377,(R4)=177777,CC=1001
3046	014626	103002	BCC	BIS3		
3047	014630	102401	BVS	BIS3		
3048	014632	100401	BMI	,+4		
3049	014634	104400	BIS1:	HLT		
3050						
3051	014636	041314	BIC	(R3),(R4)		(R3)=000377,(R4)=177400,CC=1001
3052	014640	103002	BCC	BIC3		
3053	014642	102401	BVS	BIC3		
3054	014644	100401	BMI	,+4		
3055	014646	104400	BIC1:	HLT		
3056						
3057	014650	000262	SEV			SET V
3058	014652	021314	CMP	(R3),(R4)		(R3)=000377,(R4)=177400,CC=0001
3059	014654	103003	BCC	CMP3A		
3060	014656	102402	BVS	CMP3A		
3061	014658	001401	BEQ	CMP3A		
3062	014652	100001	BPL	,+4		
3063	014664	104400	CMP1A:	HLT		
3064						
3065	014666	005013	CLR	(R3)		(R3)=000000
3066	014670	000261	SEC			
3067	014672	000013	ROR	(R3)		(R3)=100000
3068	014674	011314	MOV	(R3),(R4)		(R3)=(R4)=100000

3069	014676	005114	COM	(R4)		(R4)=077777
3070	014700	161314	SUB	(R3),(R4)		(R3)=100000,(R4)=177777,CC=1011
3071	014702	103002	BCC	SUB3A		
3072	014704	102001	BVC	SUB3A		
3073	014706	100401	BMI	,+4		
3074	014710	104400	SUB1A:	HLT		
3075						
3076	014712	000277	SCC			
3077	014714	161314	SUB	(R3),(R4)		(R3)=100000,(R4)=077777,CC=0000
3078	014716	101402	BLOS	SUB3B		BRANCH IF C OR Z IS SET
3079	014720	102401	BVS	SUB3B		
3080	014722	100001	BPL	,+4		
3081	014724	104400	SUB1B:	HLT		
3082						
3083	014726	011314	MOV	(R3),(R4)		(R3)=100000,(R4)=100000,CC=1000
3084	014730	001401	BEQ	MOV3		
3085	014732	100401	BMI	,+4		
3086	014734	104400	MOV1:	HLT		
3087						
3088	014736	061314	ADD	(R3),(R4)		(R3)=100000,(R4)=000000,CC=0111
3089	014740	103003	BCC	ADD3A		
3090	014742	102002	BVC	ADD3A		
3091	014744	001001	BNE	ADD3A		
3092	014746	100001	BPL	,+4		
3093	014750	104400	ADD1A:	HLT		
3094						
3095	014752	005113	COM	(R3)		(R3)=077777
3096	014754	011314	MOV	(R3),(R4)		(R4)=077777
3097	014756	061314	ADD	(R3),(R4)		(R3)=077777,(R4)=177776,CC=1010
3098	014760	103402	BCC	ADD3B		
3099	014762	102001	BVC	ADD3B		
3100	014764	100401	BMI	,+4		
3101	014766	104400	ADD1B:	HLT		
3102						
3103	014770	062714	ADD	#2,(R4)		
3104	014774	005714	TST	(R4)		CHECK FINAL RESULT
3105	014776	001401	BEQ	,+4		
3106	015000	104400	HLT			
3107	015002	104000	SCOPE			
3108						
3109						CHECK BINARY BYTE OPS USING ADDRESS MODE 1
3110	015004	000402	BR	,+6		
3111	015006	000000	.WORD	0		
3112	015010	000000	.WORD	0		
3113	015012	010705	MOV	PC,R5		
3114	015014	005745	TST	=(R5)		
3115	015016	005045	CLR	=(R5)		(R5)=000000
3116	015020	010902	MOV	R5,R2		
3117	015022	005042	CLR	=(R2)		(R2)=000000
3118	015024	005202	INC	R2		R2 POINTS TO ODD BYTE
3119	015026	105112	COMB	(R2)		(R2)=177400
3120						
3121	015030	000277	SCC			
3122	015032	112115	+OVB	(R2),(R5)		(R2)=177400,(R5)=000377,CC=1001

3123	015034	13005	BCC	MOVBI		
3124	015036	22404	BVS	MOVBI		
3125	015040	31403	BEQ	MOVBI		
3126	015042	100002	BPL	MOVBI		
3127	015044	105215	INCB	(R5)		;CHECK RESULT
3128	015046	001401	BEQ	,+4		
3129	015050	104400	MOVBI	HLT		
3130						
3131	015252	106312	ASLB	(R2)		;SHIFT (R2) UNTIL
3132	015054	102376	BVC	,+2		(R2)=000000
3133	015056	100912	RORB	(R2)		(R2)=100000
3134	015260	105315	DECB	(R5)		(R5)=00377
3135	015062	100015	RORB	(R5)		(R5)=000177
3136	015064	000257	CCC			
3137	015266	121512	CMPB	(R5),(R2)		(R5)=000177,(R2)=100000, CC=010
3138	015070	102001	BVC	CMPBI		
3139	015072	100401	BMI	,+4		
3140	015074	104400	CMPBI	HLT		
3141						
3142	015276	005003	CLR	R3		
3143	015100	000261	SEC			
3144	015102	000003	ROR	R3		;R3=100000
3145	015104	050315	BIS	R3,(R5)		(R5)=100177
3146	015106	000273	+SEC:SEVISEM			;SET C,V, & N
3147	015110	131215	BITB	(R2),(R5)		(R2)=100000,(R5)=100177, CC=0101
3148	015112	103002	RCC	BITBI		
3149	015114	102401	BVS	BITBI		
3150	015116	001401	BEQ	,+4		
3151	015120	104400	BITBI	HLT		
3152						
3153	015122	151215	BISB	(R2),(R5)		(R2)=100000,(R5)=100377, CC=1001
3154	015124	103001	BCC	BISBI		
3155	015126	100401	BMI	,+4		
3156	015130	104400	BISBI	HLT		
3157						
3158	015132	141215	BICB	(R2),(R5)		(R2)=100000,(R5)=100177, CC=0001
3159	015134	103002	RCC	BICBI		
3160	015136	001401	BEQ	BICBI		
3161	015140	100001	BPL	,+4		
3162	015142	104400	BICBI	HLT		
3163						
3164	015144	105112	COMB	(R2)		(R2)=077400,(R5)=100177
3165	015146	121215	CMPB	(R2),(R5)		
3166	015150	001401	BEQ	,+4		
3167	015152	104400	HLT			
3168						
3169	015154	141512	BICB	(R5),(R2)		(R5)=100177,(R2)=000000, CC=0100
3170	015156	001002	BNE	BICBIA		
3171	015160	105712	TSTB	(R2)		
3172	015162	001401	BEQ	,+4		
3173	015164	104400	BICBIA	HLT		
3174						
3175	015166	000402	BR	,+6		;RESERVE TWO WORDS FOR DATA
3176	015170	000000	WORD	0		;SOURCE DATA

3177	015172	000000	WORD	0		;DEST DATA
3178	015174	018705	MOV	PC,R5		
3179	015176	005745	TST	=(R5)		
3180	015200	105045	CLRB	=(R5)		;R5 POINTS TO DEST ODD BYTE
3181	015202	010504	MOV	R5,R4		
3182	015204	105044	CLRB	=(R4)		;R4 POINTS TO DEST EVEN BYTE
3183	015206	010403	MOV	R4,R3		
3184	015210	105043	CLRB	=(R3)		;R3 POINTS TO SOURCE ODD BYTE
3185	015212	010302	MOV	R3,R2		
3186	015214	105042	CLRB	=(R2)		;R2 POINTS TO SOURCE EVEN BYTE
3187						
3188						
3189						
3190	015216	000261	SEC			;SET CARRY
3191						
3192	015220	106112	ROLB	(R2)		(R2),(R3),(R4),(R5)
3193	015222	111214	MOVBI	(R2),(R4)		0001,0000,0000,0000
3194	015224	106112	ROLB	(R2)		0001,0000,0001,0000
3195	015226	111213	MOVBI	(R2),(R3)		0010,0000,0001,0000
3196	015230	106112	ROLB	(R2)		0010,0010,0001,0000
3197	015232	111315	MOVBI	(R3),(R5)		0010,0010,0001,0010
3198	015234	106112	ROLB	(R2)		0000,0010,0001,0010
3199	015236	106113	ROLB	(R3)		0000,0100,0001,0010
3200	015240	151215	BISB	(R2),(R5)		0000,0100,0001,0010
3201	015242	131512	BITB	(R5),(R2)		0000,0100,0001,0010
3202	015244	001426	BEQ	BIN3		
3203	015246	151314	BISB	(R3),(R4)		0000,0100,0101,0010
3204	015250	131413	BITB	(R4),(R3)		0000,0100,0101,0010
3205	015252	001423	BEQ	BIN3		
3206	015254	105213	INCB	(R3)		0000,0101,0101,0010
3207	015256	121314	CMPB	(R3),(R4)		0000,0101,0101,0010
3208	015260	001020	BNE	BIN3		
3209	015262	106113	ROLB	(R3)		0000,1010,0101,0010
3210	015264	121315	CMPB	(R3),(R5)		0000,1010,0101,0010
3211	015266	001015	BNE	BIN3		
3212	015270	106212	ASRB	(R2)		0010,1010,0101,0010
3213	015272	131214	BITB	(R2),(R4)		0010,1010,0101,0010
3214	015274	001412	BEQ	BIN3		
3215	015276	100015	RORB	(R5)		0010,1010,0101,0010
3216	015300	121415	CMPB	(R4),(R5)		0010,1010,0101,0010
3217	015302	001007	BNE	BIN3		
3218	015304	105314	DECB	(R4)		0010,1010,0100,0010
3219	015306	141214	BICB	(R2),(R4)		0010,1010,0000,0010
3220	015310	001004	RNE	BIN3		
3221	015312	111314	MOVBI	(R3),(R4)		0010,1010,0101,0010
3222	015314	106213	ASRB	(R3)		0010,0101,0101,0010
3223	015316	141315	BICB	(R3),(R5)		0010,0101,0101,0010
3224	015320	001401	BEQ	,+4		
3225	015322	104400	RINBI	HLT		
3226	015324	104000	SCOPE			
3227						
3228						
3229	015326	010405	MOV	R4,R5		;CHECK BINARY WORD OPS USING ADDRESS MODE 2 & 4
3230	015330	012715	MOV	#1,(R5)		;SET DESTINATION REGISTER

```

3231 015334 012712 1 777      MOV    #=1,(R2)
3232 015340 000257      CCC
3233 015342 000262      SEV
3234 015344 042225      ADD    (R2)+,(R5)+      ;(R2)=17777,(R5)=000000,CC=0101
3235 015346 103002      BCC
3236 015350 102401      BVS   ADD2
3237 015352 001401      BEQ   ,*4
3238 015354 104400      ADD2  HLT
3239
3240 015356 000242      SEV
3241 015360 024527 000001      CMP    =(R5),#1      ;(R5)=000000,CC=1001
3242 015364 103002      BCC   CMP2
3243 015366 102401      BVS   CMP2
3244 015370 100401      BHI   ,*4
3245 015372 104400      CMP2  HLT
3246
3247 015374 054225      BIS    =(R2),(R5)+      ;(R2)=17777,(R5)=17777,CC=1001
3248 015376 103001      RCC   B182
3249 015400 100401      BHI   ,*4
3250 015402 104400      B182 HLT
3251 015404 000277      SCC
3252 015406 000244      CLF
3253 015410 142245      SUB    (R2)+,(R5)      ;(R2)=17777,(R5)=000000,CC=0100
3254 015412 103402      BCS   SUB2
3255 015414 102401      BVS   SUB2
3256 015416 001401      BEQ   ,*4
3257 015420 104400      SUB2  HLT
3258
3259 015422 005442      NEB    =(R2)      ;(R2)=000001
3260 015424 005115      COM    (R5)      ;(R5)=17777
3261 015426 000277      SCC
3262 015430 000250      CLM
3263 015432 042225      BIC    (R2)+,(R5)+      ;(R2)=000001,(R5)=17777,CC=1001
3264 015434 103003      BCC   BIC2
3265 015436 102402      BVS   BIC2
3266 015440 001401      BEQ   BIC2
3267 015442 100401      BHI   ,*4
3268 015444 104400      BIC2  HLT
3269
3270 015446 012742 125252      MOV    #125252,-(R2)
3271 015452 012245      MOV    (R2)+,(R5)
3272 015454 005125      COM    (R5)+      ;(R5)=052525
3273 015456 000262      SEV
3274 015460 032445      BIT    =(R2),=(R5)      ;(R2)=125252,(R5)=052525,CC=0101
3275 015462 103002      BTF   BIT2
3276 015464 102401      BVS   BIT2
3277 015466 001401      BEQ   ,*4
3278 015470 104400      BIT2  HLT
3279
3280 015472 000262      SEV
3281 015474 052225      BIS    (R2)+,(R5)+      ;(R2)=125252,(R5)=17777,CC=1001
3282 015476 103002      RCC   B182A
3283 015500 102401      BVS   B182A
3284 015502 100401      BHI   ,*4
    
```

```

3285 015504 104400      B182A HLT
3286
3287 015506 042745 125252      BIC    #125252,-(R5)      ;(R5)=052525
3288 015512 005125      COM    (R5)+      ;(R5)=125252
3289 015514 024245      CMP    =(R2),-(R5)
3290 015516 001401      BEQ   ,*4
3291 015520 104400      HLT
3292
3293 015522 005012      CLR    (R2)
3294 015524 005122      COM    (R2)+      ;(R2)=17777
3295 015526 102742 000001      SUB    #1,-(R2)      ;(R2)=17776,CC=1000
3296 015532 103402      BCS   SUB2A
3297 015534 102401      BVS   SUB2A
3298 015536 100401      BHI   ,*4
3299 015540 104400      SUB2A HLT
3300 015542 104000      SCOPE
3301
3302 015544 010702      MOV    PC,R2      ;GET CURRENT PC
3303 015546 010205      MOV    R2,R5      ;MOVE TO R5
3304 015550 124245      CMPB   =(R2),-(R5)      ;COMPARE ALL PREVIOUS MEMORY ADDRESSES
3305 015552 001401      BEQ   ,*4
3306 015554 104400      HLT
3307 015556 020237 001010      CMP    R2,#PRSTAD      ;ERROR!
3308 015562 001372      BNE   ,*4      ;CHECK FOR LOW LIMIT
3309 015564 104000      SCOPE
3310
3311
3312 015566 000402      ;CHECK BINARY BYTE OPS USING ADDRESS MODES 2 & 4,
3313 015570 000000      RR    ,*6      ;RESERVE TWO WORDS
3314 015572 000000      ^WORD 0      ;SOURCE DATA
3315 015574 010703      ^WORD 0      ;DESTINATION DATA
3316 015576 005743      MOV    PC,R3
3317      TST   =(R2)
3318
3319 015600 010300      ;FIRST CHECK AUTO INCREMENT/DECREMENT
3320      MOV    R3,R0
3321 015602 010002      MOV    R0,R2
3322 015604 005302      DEC    R2
3323 015606 010604      MOV    SP,R4
3324 015610 010605      MOV    SP,R5
3325 015612 005745      TST   =(R2)
3326
3327 015614 114046      MOVB   =(R0),-(SP)
3328 015616 020506      CMP    R5,SP
3329 015620 001021      BNE   BINB
3330 015622 020200      CMP    R2,R0
3331 015624 001017      BNE   BINB
3332 015626 122026      CMPB   (R0)+,(SP)+
3333 015630 020406      CMP    R4,SP
3334 015632 001014      BNE   BINB
3335 015634 020003      CMP    R0,R3
3336 015636 001012      BNE   BINB
3337 015640 154640      B18B   -(SP),-(R0)
3338 015642 020506      CMP    R5,SP
3339 015644 001027      BNE   BINB
    
```

3339	015646	022200		CMP	R2,R0	
3340	015650	001005		BNE	BINB	
3341	015652	142620		BICB	(SP)+,(R0)+	
3342	015654	020400		CMP	R4,SP	
3343	015656	001002		BNE	BINB	
3344	015660	020003		CMP	R0,R3	
3345	015662	001401		BEQ	,+4	
3346	015664	104400		HLT		
3347	015666	104000		SCOPE		
3348						
3349	015670	010023		MOV	R0,R3	
3350	015672	112743	000200	MOV	#200,(R3)	
3351	015676	112743	000377	MOV	#377,(R3)	(R3)=100377
3352	015702	010304		MOV	R3,R4	
3353	015704	112744	000177	MOV	#177,(R4)	
3354	015710	112744	000000	MOV	#0,(R4)	(R4)=077400
3355	015714	001401		BEQ	,+4	
3356	015716	104400		HLT		
3357						
3358	015720	152324		RISB	(R3)+,(R4)+	(R3)=100377,(R4)=077777
3359	015722	100401		BMI	,+4	
3360	015724	104400		HLT		
3361						
3362	015726	122324		CMPB	(R3)+,(R4)+	
3363	015730	103402		BCC	CMPB2	
3364	015732	102001		BVC	CMPB2	
3365	015734	100001		RPL		
3366	015736	104400		CMPB2	HLT	
3367						
3368	015740	000261		SEC		
3369	015742	134344		BITB	=(R3),-(R4)	
3370	015744	103002		BCC	BITB2	
3371	015746	102401		BVS	BITB2	
3372	015750	001401		BEQ	,+4	
3373	015752	104400		BITB2	HLT	
3374						
3375	015754	000244		CLZ		
3376	015756	144344		BICB	=(R0),=(R4)	(R3)=100377,(R4)=077400
3377	015760	001401		BEQ	,+4	
3378	015762	104400		HLT		
3379	015764	104000		SCOPE		
3380						
3381						
3382	015766	000404		BR	2\$;CHECK BINARY WORD QPS USING ADDRESS MODES 3 & 5, ;RESERVE SPACE FOR DATA AND ADDRESSES
3383	015770	000000		WORD	0	;CONTAINS ADDRESS OF SOURCE DATA
3384	015772	000000		WORD	0	;CONTAINS ADDRESS OF DEST DATA
3385	015774	000000		WORD	0	;CONTAINS SOURCE DATA
3386	015776	000000		WORD	0	;CONTAINS DEST DATA
3387	016000	010701		2\$	MOV	PC,R1
3388	016002	010100		MOV	R1,R0	;SET SCOPE PTR
3389	016004	024040		CMP	=(R0),=(R0)	;ADJUST R0
3390	016006	010005		MOV	R0,R5	;R5 POINTS TO DEST DATA
3391	016010	024545		CMP	=(R5),=(R5)	;SUB 4 FROM R5
3392	016012	010015		MOV	R0,(R5)	;R5 POINTS TO ADDRESS OF DEST DATA

3393	016014	010502		MOV	R5,R2	
3394	016016	010004		MOV	R0,R4	;R4 POINTS TO DEST DATA
3395	016020	005740		TST	=(R0)	
3396	016022	010003		MOV	R0,R3	;R3 POINTS TO SOURCE DATA
3397	016024	010042		MOV	R0,(R2)	;R2 POINTS TO ADDRESS OF SOURCE DATA
3398	016026	005013		CLR	(R3)	;PRESET SOURCE DATA
3399	016030	005014		CLR	(R4)	;PRESET DEST DATA
3400						
3401	016032	000277		SCC		
3402	016034	000244		CLZ		
3403	016036	163235		SUB	0,(R2)+,0,(R5)+	(R3)=000000,(R4)=000000,CC=0100
3404	016040	103402		BCC	SUB0	
3405	016042	102401		BVS	SUB3	
3406	016044	001401		BEQ	,+4	
3407	016046	104400		SUB3	HLT	
3408						
3409	016050	052752	100000	BIS	#100000,0,(R2)	(R3)=100000
3410	016054	062755	000001	ADD	#1,0,(R5)	(R4)=000001
3411	016060	163235		SUB	0,(R2)+,0,(R5)+	(R3)=100000,(R4)=100001,CC=1011
3412	016062	103002		BCC	SUB3A	
3413	016064	102001		BVC	SUB3A	
3414	016066	100401		BMI	,+4	
3415	016070	104400		SUB3A	HLT	
3416						
3417	016072	005414		NEG	(R4)	(R4)=077777
3418	016074	035255		RIT	0-(R2),0-(R5)	(R3)=100000,(R4)=077777
3419	016076	001401		BEQ	,+4	
3420	016100	104400		HLT		
3421	016102	023235		CMP	0,(R2)+,0,(R0)+	
3422	016104	102401		BVS	,+4	
3423	016106	104400		HLT		
3424	016110	005152		COM	0-(R2)	
3425	016112	000257		CCC		
3426	016114	063255		ADD	0,(R2)+,0-(R5)	
3427	016116	102001		BVC	ADD3	
3428	016120	100401		BMI	,+4	
3429	016122	104400		ADD3	HLT	
3430	016124	000261		SEC		
3431	016126	045235		BIC	0-(R2),0,(R5)+	(R3)=077777,(R4)=100000
3432	016130	103001		BCC	BICB	
3433	016132	100401		BMI	,+4	
3434	016134	104400		BIC3	HLT	
3435						
3436	016136	005155		COM	0-(R5)	(R4)=077777
3437	016140	023235		CMP	0,(R2)+,0,(R5)+	(R3)=077777,(R4)=077777
3438	016142	001401		BEQ	,+4	
3439	016144	104400		HLT		
3440	016146	104000		SCOPE		
3441						
3442						
3443	016150	000406		BR	1\$;CHECK BINARY BYTE QPS USING ADDRESS MODES 3 & 5, ;RESERVE SPACE FOR ADDRESSES & DATA
3444	016152	000000		WORD	0	;CONTAINS ADDRESS OF SOURCE DATA (EVEN BYTE)
3445	016154	000000		WORD	0	;CONTAINS ADDRESS OF SOURCE DATA (ODD BYTE)
3446	016156	000000		WORD	0	;CONTAINS ADDRESS OF DEST DATA (EVEN BYTE)

DCQKCD	11/40-11/45 CPU EXERCISER	MACY11 27(455)	4-SEP-74 11153	PAGE 71
DCQKCD	START OF SECTION 2			
3447	016160	000000		
3448	016162	000000		
3449	016164	000000		
3450				
3451	016166	010700	151	MOV PC,R0
3452	016170	024040		CMP =(R0),=(R2)
3453	016172	010003		MOV R0,R3
3454	016174	010305		MOV R3,R5
3455	016176	005743		TST =(R3)
3456	016200	010043		MOV R0,=(R3)
3457	016202	005213		INC (R3)
3458	016204	010043		MOV R0,=(R3)
3459	016206	010304		MOV R3,R4
3460	016210	005740		TST =(R0)
3461	016212	010044		MOV R0,=(R4)
3462	016214	005214		INC (R4)
3463	016216	010044		MOV R0,=(R4)
3464				
3465	016220	000261		SEC
3466	016222	012734	177001	MOV #177001,=(R4)+
3467	016226	112734	000200	MOVB #200,=(R4)+
3468	016232	115433		MOVB =(R4),=(R3)+
3469	016234	115433		MOVB =(R4),=(R3)+
3470	016236	103401		BCS ,+4
3471	016240	104400		HLT
3472	016242	022715	000600	CMP #600,(R5)
3473	016246	001401		BEQ ,+4
3474	016250	104400		HLT
3475	016252	024343		CMP =(R3),=(R3)
3476	016254	153433		BISB =(R0),=(R3)+
3477	016256	153433		BISB =(R0),=(R3)+
3478	016260	022715	100601	CMP #100601,=(R5)
3479	016264	001401		BEQ ,+4
3480	016266	104400		HLT
3481	016270	145453		BICB =(R4),=(R3)
3482	016272	145453		BICB =(R4),=(R3)
3483	016274	133433		BITB =(R0),=(R3)+
3484	016276	001002		BNE BITB3
3485	016300	135433		BITB =(R4),=(R3)+
3486	016302	001001		BNE ,+4
3487	016304	104400		BITB31 HLT
3488				
3489	016306	123453		CMPB =(R4)+,=(R3)
3490	016310	001002		BNE CMPB3

DCQKCD	11/40-11/45 CPU EXERCISER	MACY11 27(455)	4-SEP-74 11153	PAGE 72
DCQKCD	START OF SECTION 2			
3491	016312	123453		
3492	016314	001401		
3493	016316	104400		
3494	016320	104000		
3495				
3496				
3497	016322	000402		
3498	016324	000000		
3499	016326	000000		
3500				
3501	016330	013702	001004	MOV #F6CYOR,R2
3502	016334	010205		MOV R2,R5
3503	016336	005065	016326	CLR DDATA(5)
3504	016342	012762	000001 016324	MOV #1,SDATA(2)
3505	016350	050265	016324 016326	151 SDATA(2),DDATA(5)
3506	016356	006362	016324	ASL SDATA(2)
3507	016362	006362	016324	ASL SDATA(2)
3508	016366	103370		BCC 15
3509	016370	022765	052525 016326	CMP #52525,DDATA(5)
3510	016376	001401		BEQ ,+4
3511	016400	104400		HLT
3512	016402	012762	177777 016324	MOV #1,SDATA(2)
3513	016410	040562	016326 016324	RIC DDATA(5),SDATA(2)
3514	016416	030265	016324 016326	RIT SDATA(2),DDATA(5)
3515	016424	001401		BEQ ,+4
3516	016426	104400		HLT
3517	016430	006365	016326	ASL DDATA(5)
3518	016434	020265	016324 016326	CMP SDATA(2),DDATA(5)
3519	016442	001401		REQ ,+4
3520				
3521	016444	104400		HLT
3522	016446	000257		CCC
3523	016450	060265	016324 016326	ADD SDATA(2),DDATA(5)
3524	016456	103002		BCC ADD6
3525	016460	102001		BVC ADD6
3526	016462	100001		RPL ,+4
3527	016464	104400		HLT
3528				
3529	016466	006362	016324	ASL SDATA(2)
3530	016472	160265	016324 016326	SUB SDATA(2),DDATA(5)
3531	016500	103401		RCS SUR6
3532	016502	001401		BEQ ,+4
3533	016504	104400		SUB61 HLT
3534				
3535	016506	112700	000377	MOVB #377,R0
3536	016512	010062	016324	MOV R0,SDATA(2)
3537	016516	012765	177777 016326	MOV #1,DDATA(5)
3538	016524	166500	016326	SUB DDATA(5),R0
3539	016530	001401		REQ ,+4
3540	016532	104400		HLT
3541	016534	060265	016324 016326	151 ADD SDATA(2),DDATA(5)
3542	016542	006362	016324	ASL SDATA(2)
3543	016546	005162	016324	COM SDATA(2)
3544	016552	030265	016324 016326	RIT SDATA(2),DDATA(5)

3545	016560	001401				BEQ	,+4	
3546	016562	104400				HLT		
3547	016564	005162	016324			COM	SDATA(2)	
3548	016570	026265	016324	016326		CMP	SDATA(2),DDATA(5)	
3549	016576	001401				BEQ	,+4	
3550	016600	104400				HLT		
3551	016602	026200	016324			CMP	SDATA(2),R0	
3552	016606	001352				BNE	15	
3553	016610	104000				SCOPE		
3554								
3555								
3556								
3557								
3558								
3559	016612	013702	001004			MOV	#FACTOR,R2	IGET INDEX VALUE
3560	016616	010204				MOV	R2,R4	IR2 FOR SOURCE EVEN BYTE INDEX, R4 FOR
3561	016620	010403				MOV	R4,R3	DEST ODD BYTE, R3 FOR SOURCE EVEN
3562	016622	005203				INC	R3	AND R5 FOR DEST ODD BYTE
3563	016624	010305				MOV	R3,R5	
3564	016626	000261				SEC		ISSET CARRY
3565	016630	012702	125252	016754		MOV	#125252,SDATAB(2)	
3566	016636	112703	177125	016754		MOV	#177125,SDATAB(3)	ISOURCE DATA = 002652
3567	016644	016264	016754	016756		MOV	SDATAB(2),DDATAB(4)	
3568	016652	052764	125125	016756		BIS	#125125,DDATAB(4)	IDEST DATA = 177777
3569	016660	136203	016754	016754		BITB	SDATAB(2),SDATAB(3)	
3570	016666	001401				BEQ	,+4	
3571	016670	104400				BITB6	HLT	
3572								
3573	016672	146264	016754	016756		BICB	SDATAB(2),DDATAB(4)	
3574	016700	103401				BCS	,+4	
3575	016702	104400				HLT		IFERROR MOV,BIS,BITBIC DO NOT AFFECT 'C'
3576	016734	126364	016754	016756		CMPB	SDATAB(3),DDATAB(4)	
3577	016712	001401				BEQ	,+4	
3578	016714	104400				HLT		
3579								
3580	016716	146365	016754	016756		BICB	SDATAB(3),DDATAB(5)	
3581	016724	126244	016754	016756		CMPB	SDATAB(2),DDATAB(5)	
3582	016732	001401				BEQ	,+4	
3583	016734	104400				HLT		
3584								
3585	016736	136564	016756	016756		BITB	DDATAB(5),DDATAB(4)	
3586	016744	001401				BEQ	,+4	
3587	016746	104400				HLT		
3588	016750	104000				SCOPE		
3589								
3590	016752	000406				BR	UB7	RESERVE TWO WORDS
3591	016754	000000				SDATAB	.WORD 0	RESERVED FOR SOURCE DATA
3592	016756	000000				DDATAB	.WORD 0	RESERVED FOR DEST DATA
3593								
3594								
3595								
3596	016760	000000				IR2=ADDRESS OF SOURCE DATA, AND R3= ADDRESS OF DEST DATA		
3597	016762	000000				SBIN7	.WORD 0	CONTAINS ADDRESS OF SOURCE DATA
3598	016764	000000				DBIN7	.WORD 0	CONTAINS ADDRESS OF DEST DATA
							.WORD 0	CONTAINS SOURCE DATA

3599	016766	000000				.WORD	0	CONTAINS DEST DATA
3600								
3601	016770	010700				UB7	MOV	PC,R0
3602	016772	024040				CMP	=(R0),-(R0)	
3603	016774	010002				MOV	R0,R2	
3604	016776	024242				CMP	=(R2),-(R2)	
3605	017000	010012				MOV	R0,(R2)	
3606	017002	010203				MOV	R2,R3	
3607	017004	024043				CMP	=(R0),-(R3)	
3608	017006	010013				MOV	R0,(R3)	
3609								
3610	017010	000261				SEC		
3611	017012	012777	100000	177740		MOV	#100000,SBIN7	ISOURCE DATA = 100000
3612	017020	017777	177734	177734		MOV	SBIN7,DBIN7	IDEST DATA = 100000
3613	017026	103001				BCC	MOV7	
3614	017030	100401				BMI	,+4	
3615	017032	104400				MOV7	HLT	
3616	017034	006377	177722			ASL	DBIN7	IDEST DATA = 000000
3617	017040	102001				BVC	,+4	
3618	017042	001401				BEQ	,+4	
3619	017044	104400				HLT		
3620								
3621	017046	027777	177706	177706		CMP	SBIN7,DBIN7	(R2)=100000,(R3)=000000
3622	017054	103402				BCS	CMP7	
3623	017056	102401				BVS	CMP7	
3624	017060	100401				BMI	,+4	
3625	017062	104400				CMP7	HLT	
3626								
3627	017064	167777	177670	177670		SUB	SBIN7,DBIN7	(R2)=100000,(R3)=100000
3628	017072	103003				BCC	SUB7	
3629	017074	102002				BVC	SUB7	
3630	017076	001401				BEQ	SUB7	
3631	017100	100401				BMI	,+4	
3632	017102	104400				SUB7	HLT	
3633								
3634	017104	006277	177650			ASR	SBIN7	(R2)=140000
3635	017110	067777	177644	177644		ADD	SBIN7,DBIN7	(R2)=140000,(R3)=040000
3636	017116	103003				BCC	ADD7	
3637	017120	102002				BVC	ADD7	
3638	017122	001401				BEQ	ADD7	
3639	017124	100001				BPL	,+4	
3640	017126	104400				ADD7	HLT	
3641								
3642	017130	047777	177624	177624		BIC	SBIN7,DBIN7	(R2)=140000,(R3)=000000
3643	017136	001401				BEQ	,+4	
3644	017140	104400				HLT		
3645								
3646	017142	057777	177612	177612		RIS	SBIN7,DBIN7	(R2)=140000,(R3)=140000
3647	017150	100401				BMI	,+4	
3648	017152	104400				HLT		
3649								
3650	017154	027777	177600	177600		CMP	SBIN7,DBIN7	
3651	017162	001401				BEQ	,+4	
3652	017164	104400				HLT		

3653 017166 104000
3654
3655
3656
3657 017170 005000
3658 017172 005067 000072
3659 017176 010707
3660 017200 120707
3661 017202 030707
3662 017204 060007
3663 017206 105707
3664 017210 005507
3665 017212 021007
3666 017214 131007
3667 017216 062707 000000
3668 017222 023707 001004
3669 017226 133707 001004
3670 017232 000240
3671
3672
3673 017234 163707 001004
3674 017240 063707 001004
3675 017244 000240
3676 017246 024607
3677 017250 132607
3678 017252 024707 000012
3679 017256 166707 000006
3680 017262 046707 000002
3681 017266 000401
3682 017270 000000
3683 017272 104000
3684
3685 017274 010702
3686 017276 062702 000012
3687 017302 012707 001152
3688 017306 000000
3689
3690
3691
3692
3693
3694 017310 010700
3695 017312 005740
3696 017314 010037 001010
3697 017320 012737 000003 005176
3698 017326 004737 005166
3699 017332 013767 005172 002104
3700 017340 010700
3701 017342 162700 017342
3702 017346 010037 001004
3703 017352 010701
3704
3705
3706 017354 012703 125252

SCOPE
ISOME MISCELLANEOUS OPERATION INVOLVING THE PC
(NOTE) NONE OF THESE OPERATIONS SHOULD AFFECT THE PC
CLR R0
CLR R1
MOV PC,PC
CHPB PC,PC
BIT PC,PC
ADD R0,PC
TSTB PC
ADC PC
CMP (R0),PC
BITB (R0),PC
ADD #0,PC
CMP #FACTOR,PC
BITB #FACTOR,PC
NOP
;THE NEXT TWO INSTRUCTION CAUSE THE PROGRAM TO JUMP TO THE UNRELOCATED
;CODE AND TO RETURN ON THE FOLLOWING INST (IF THE CODE IS RELOCATED)
SUB #FACTOR,PC ;JUMPS TO UNRELOCATED CODE
ADD #FACTOR,PC ;RETURNS
NOP
CMP -(SP),PC
BITB (SP),PC
CMP R1,PC
SUB R1,PC
BIC R1,PC
BR ,+4 ;BRANCH OVER 15
151 0
SCOPE
MOV PC,R2
ADD #12,R2
MOV #RELOC,PC ;GO RELOCATE PROGRAM CODE
REL221: .WORD 0
;22222222222222 LAST ADDRESS OF CODE TO BE RELOCATED 2222222222
;SYTL START OF SECTION 3
13333333333333 FIRST ADDRESS TO BE RELOCATED 3333333333
REL31: MOV PC,R0 ;GET PC
TST R0 ;R0 CONTAINS THE ADDRESS OF REL3
MOV R0,#FFRSTAD ;SAVE
MOV #3,#SECT ;SET SECTION #
JSR PC,#LDDISP ;LOAD DISPLAY GEG
MOV #DISPLY,REL33
MOV PC,R0 ;GET CURRENT PC
SUB #1,R0 ;SUBTRACT RELOCATION FACTOR
MOV R0,#FACTOR ;SAVE RELOCATION FACTOR
MOV PC,R1 ;SET NEW SCOPE PTR
;CHECK BINARY BYTE OPS USING ADDRESS MODE 0
MOV #129252,R0

3707 017360 010304
3708 017362 140304
3709 017364 022704 125000
3710 017370 001401
3711 017372 104400
3712
3713 017374 005004
3714 017376 150304
3715 017400 022704 000252
3716 017404 001401
3717 017406 104400
3718
3719 017410 110404
3720 017412 022704 177652
3721 017416 001401
3722 017420 104400
3723
3724 017422 132704 177525
3725 017426 001401
3726 017430 104400
3727
3728 017432 105104
3729 017434 110404
3730 017436 022704 000125
3731 017442 001401
3732 017444 104400
3733
3734 017446 150304
3735 017450 105204
3736 017452 001401
3737 017454 104400
3738 017456 104000
3739
3740
3741 017460 000406
3742 017462 000000
3743 017464 000000
3744 017466 000000
3745 017470 000000
3746 017472 000000
3747 017474 000000
3748
3749 017476 010700
3750 017500 024040
3751 017502 010060 177774
3752 017506 010060 177774
3753 017512 005260 177774

MOV R3,R4 ;R3=R4=125252
BICB R3,R4 ;R3=125252,R4=125000
CMP #129000,R4 ;CHECK RESULT
BEQ ,+4
HLT
CLR R4 ;R3=125252,R4=0
BITB R3,R4 ;R3=125252,R4=000252
CMP #250,R4 ;CHECK RESULT
BEQ ,+4
HLT
MOVB R4,R4 ;R4=177652
CMP #177652,R4 ;CHECK RESULT
REQ ,+4
HLT
BITB #177525,R4
BEQ ,+4
HLT
COMB R4 ;R4=177525
MOVB R4,R4 ;R4=000125
CMP #125,R4 ;CHECK RESULT
BEQ ,+4
HLT
BISB R3,R4 ;R3=125252,R4=000377
INCB R4
BEQ ,+4
HLT
SCOPE
;CHECK BINARY BYTE OPS USING ADDRESS MODE 7
BR BINB7 ;RESERVE SPACE FOR ADDRESSES & DATA
;BINB7: .WORD 0 ;CONTAINS ADDRESS OF SOURCE EVEN BYTE
; .WORD 0 ;CONTAINS ADDRESS OF SOURCE ODD BYTE
; .WORD 0 ;CONTAINS ADDRESS OF DEST EVEN BYTE
; .WORD 0 ;CONTAINS ADDRESS OF DEST ODD BYTE
DBINB7: .WORD 0 ;CONTAINS SOURCE DATA
; .WORD 0 ;CONTAINS DEST DATA
BINB7: MOV PC,R0
;-(R0),-(R0) ;R0 = ADDRESS OF DEST DATA
R0,-6(R0) ;LOAD ADDRESS OF DEST EVEN BYTE DATA
MOV R0,-4(R0)
INC =4(R0) ;LOAD ADDRESS OF DEST ODD BYTE DATA

DC0KCD 11/40-11/45 CPU EXERCISER MACY11 27(655) 4-SEP-74 1153 PAGE 77
 DC0KCD START OF SECTION 3

3754	017516	005740		TST	=(R0)	JR0=ADDRESS OF SOURCE DATA
3755	017520	010060	177770	MOV	R0,=10(R0)	JLOAD ADDRESS OF SOURCE EVEN BYTE DATA
3756	017524	010060	177772	MOV	R0,=6(R0)	
3757	017530	005260	177772	INC	=(R0)	JLOAD ADDRESS OF SOURCE ODD BYTE DATA
3758						
3759	017534	005002		CLR	R2	JSET INDEX REGISTERS
3760	017536	012703	000002	MOV	#2,R3	J#SBINB7(2)J#SBINB7(3) REFERENCE EVEN &
3761	017542	012704	177774	MOV	#4,R4	JODD BYTE SOURCE DATA J#DBINB7(4)J#DBINB7(5)
3762	017546	012705	177776	MOV	#-2,R5	JREFERENCE DEST EVEN& ODD BYTE DATA
3763						
3764						
3765	017552	005020		CLR	(R0)+	JPRESET SOURCE DATA
3766	017554	005010		CLR	(R0)	JPRESET DEST DATA
3767	017556	013746	001004	MOV	#F&CYOR,=(SP)	JGET RELOCATION FACTOR
3768	017562	001602		ADD	(SP),R2	JAND ADD TO INDEX VALUES
3769	017564	001603		ADD	(SP),R3	
3770	017566	001604		ADD	(SP),R4	
3771	017570	002605		ADD	(SP)+,R5	
3772						
3773	017572	112773	177777	MOVB	#-1,#SBINB7(3)	JSRC DATA = 177400
3774	017600	132772	000377	BITB	#377,#SBINB7(2)	JCHECK THAT EVEN BYTE WAS NOT AFFECTED
3775	017606	001401		BEQ	,+4	JBY MOVB INSTRUCTION
3776	017610	104400		HLT		
3777						
3778	017612	157374	017462	BISB	#SBINB7(3),#DBINB7(4)	
3779	017620	105274	017472	INCB	#DBINB7(4)	JCHECK THAT BIS SET ALL BITS
3780	017624	001401		BEQ	,+4	
3781	017626	104400		HLT		
3782						
3783	017630	105375	017472	DECB	#DBINB7(5)	JDEST DATA = 177400
3784	017634	005274	017472	INC	#DBINB7(4)	JDEST DATA = 177401
3785	017640	127375	017462	CMPB	#SBINB7(3),#DBINB7(5)	
3786	017646	001401		BEQ	,+4	
3787	017650	104400		HLT		
3788						
3789	017652	147375	017462	BICB	#SBINB7(3),#DBINB7(5)	
3790	017660	001401		BEQ	,+4	
3791	017662	104400		HLT		
3792						
3793	017664	105073	017462	CLRB	#SBINB7(3)	JSRC DATA = 000000
3794						JTHIS ROUTINE SETS ALL BITS IN THE SOURCE ODD BYTE BY BISING A BIT FROM
3795						JTHE DEST EVEN BYTE INTO THE SOURCE ODD BYTE
3796	017670	157473	017472	BISB	#DBINB7(4),#SBINB7(3)	
3797	017676	106174	017472	ROLB	#DBINB7(4)	
3798	017702	103372		BCC	BISB	
3799	017704	022772	177400	CMP	#177400,#SBINB7(2)	JCHECK RESULT
3800	017712	001401		BEQ	,+4	
3801	017714	104400		HLT		
3802						
3803	017716	000372	017462	SWAB	#SBINB7(2)	JSRC DATA = 000377
3804	017722	12775	000200	MOVB	#200,#DBINB7(5)	JDEST DATA = 100000
3805						
3806	017730	147572	017472	BICB	#DBINB7(5),#SBINB7(2)	
3807	017736	106075	017472	RORB	#DBINB7(5)	

DC0KCD 11/40-11/45 CPU EXERCISER MACY11 27(655) 4-SEP-74 1153 PAGE 78
 DC0KCD START OF SECTION 3

3808	017742	103372		BCC	BICB	
3809	017744	005772	017462	TST	#SBINB7(2)	
3810	017750	001401		BEQ	,+4	
3811	017752	104400		HLT		
3812	017754	104000		SCOPE		
3813						
3814	017756	012702	000001	OAERR	R2	JLOAD R2 WITH ODD #
3815	017762	010703		MOV	PC,R3	
3816	017764	000401		BR	,+4	JRESERVE SPACE FOR A WORD
3817	017766	000000		WORD	0	JWILL CONTAIN AN ODD ADDRESS
3818	017770	005723		TST	(R3)+	JSTEP R3 TO POINT TO WORD ABOVE
3819	017772	010313		MOV	R3,(R3)	
3820	017774	005213		INC	(R3)	JAND MAKE ODD
3821	017776	012737	020124	MOV	#13,#ERRVEC	JSET ODD ADDRESS & RESERVED INSTRUCTION
3822	020004	063737	001004	ADD	#FACTOR,#ERRVEC	
3823	020012	013737	000004	MOV	#ERRVEC,#RESVEC	JTO TRAP TO 15 BELOW
3824						
3825	020020	000277		SCC		JSET ALL CC'S
3826	020022	100212		SUB	R2,(R2)	
3827	020024	104400		HLT		
3828	020026	000222		ADD	R2,(R2)+	
3829	020030	104400		HLT		
3830	020032	000342		ASL	=(R2)	
3831	020034	104400		HLT		
3832	020036	100512		MFPD	(R2)	JNOTE! MAY BE RESERVED
3833	020040	104400		HLT		
3834	020042	170412		CLRF	(R2)	
3835	020044	104400		HLT		
3836	020046	042202		BIC	(R2)+,R2	
3837	020050	104400		HLT		
3838	020052	104272		SUB	-(R2),R2	
3839	020054	104400		HLT		
3840	020056	155202		BISB	#=(R2),R2	
3841	020060	104400		HLT		
3842	020062	105532		ADCB	#(R2)+	
3843	020064	104400		HLT		
3844	020066	103302		SUB	#(R3)+,R2	
3845	020070	104400		HLT		
3846	020072	005733		TST	#(R0)+	
3847	020074	104400		HLT		
3848	020076	100533		MFPD	#(R0)+	
3849	020100	104400		HLT		
3850	020102	170453		CLRD	#=(R3)	
3851	020104	104400		HLT		
3852	020106	137702	177775	BITB	#,+3,R2	
3853	020112	104400		HLT		
3854	020114	105477	177773	NEGB	#,+3	
3855	020120	104400		HLT		
3856	020122	000406		RR	25	
3857						
3858	020124	062716	000002	ADD	#2,(SP)	JADJUST RETURN PC
3859	020130	052766	000017	RIS	#17,2(SP)	JSET CONDITION CODES ON RETURN
3860	020136	000002		RTI		
3861						

```

3862 020140 012700 000500 25; MOV #STKPTR,SP ;RESET STACK PTR
3863 020144 012737 005274 000004 MOV #ERRPT,#ERRVEC ;RESET TIME OUT VECTOR
3864 020152 012737 009264 000010 MOV #RESEBR,#RESVEC
3865 020160 104000 SCOPE
3866
3867 ICHECK JMP INSTRUCTIONS
3868
3869 020162 010700 MOV PC,R0
3870 020164 062700 000012 ADD #12,R0 ;SET ADDRESS FOR JMP INST
3871 020170 000277 SCC ;SET CC'S
3872 020172 000110 JMP (R0)
3873 020174 000402 BR ,+6
3874 020176 000250 CLN ;JMP INST JUMPS HERE
3875 020200 000775 BR ,+4
3876
3877 020202 103003 BCC JMP1
3878 020204 102002 BVC JMP2
3879 020206 001001 BNE JMP3
3880 020210 100001 BPL
3881 020212 104400 JMP1; HLT ;ERROR! INCORRECT CC'S AFTER JMP
3882
3883 020214 005002 CLR R2 ;SET INDICATOR
3884 020216 010703 MOV PC,R3
3885 020220 000401 BR ,+4 ;RESERVE WORD FOR JMP ADDRESS
3886 020222 000000 ;WORD 0 ;CONTAINS ADDRESS FOR JMP INST
3887 020224 005723 TST (R3)+
3888 020226 010313 MOV R3,(R3)
3889 020230 010300 MOV R3,R0
3890 020232 062713 ADD #27,(R3) ;(R3) IS JMP ADDRESS
3891 020236 010300 MOV R3,R0
3892 020240 000133 JMP @(R3); ;JUMP TO ADDRESS CONTAINED IN R3
3893 020242 000402 BR ,+6
3894 020244 005102 COM R2 ;COMPLEMENT INDICATOR
3895 020246 000775 BR ,+4
3896 020250 005202 INC R2 ;CHECK INDICATOR
3897 020252 001003 BNE JMP3
3898 020254 005720 TST (R0)+
3899 020256 020003 CMP R0,R3 ;CHECK AUTO-INC R3
3900 020260 001401 BEQ
3901 020262 104400 JMP3; HLT
3902
3903 020264 005002 CLR R2 ;SET INDICATOR
3904 020266 010704 MOV PC,R4
3905 020270 010400 MOV R4,R0 ;SET UP CHECK REGISTER
3906 020272 000402 BR 1$
3907 020274 005102 COM R2 ;COMPLEMENT INDICATOR
3908 020276 000403 BR 2$
3909 020300 022424 1$; CMP (R4+),(R4)+
3910 020302 005724 TST (R4)+ ;R4=JMP ADDRESS
3911 020304 000144 JMP *(R4) ;USE R4 AS ADDRESS
3912 020306 005202 INC R2 ;CHECK INDICATOR
3913 020310 001003 BNE JMP4
3914 020312 022020 CMP (R0+),(R0)+
3915 020314 020004 CMP R0,R4 ;CHECK AUTO-DEC R4

```

```

3916 020316 001401 BEQ ,+4
3917 020320 104400 JMP4; HLT
3918
3919 020322 010703 MOV PC,R3
3920 020324 000401 BR ,+4 ;RESERVE WORD FOR JMP ADDRESS
3921 020326 000000 ;WORD 0 ;CONTAINS JUMP ADDRESS
3922 020330 005723 TST (R3)+
3923 020332 010313 MOV R3,(R3)
3924 020334 062723 ADD #16,(R3)+
3925 020340 010300 MOV R3,R0 ;LOAD CHECK REGISTER
3926 020342 000402 BR 3$
3927 020344 005102 COM R2
3928 020346 000401 BR 4$
3929 020350 000153 3$; JMP *(R3) ;JUMP TO 2$ VIA 1$ ABOVE
3930 020352 005202 INC R2 ;CHECK INDICATOR
3931 020354 001003 BNE JMP5
3932 020356 005740 TST *(R0)
3933 020360 020003 CMP R0,R3 ;CHECK AUTO-DEC R3
3934 020362 001401 BEQ
3935 020364 104400 JMP5; HLT
3936
3937 020366 000402 BR 2$
3938 020370 005102 COM R0 ;COMPLEMENT INDICATOR
3939 020372 000402 BR 3$
3940 020374 000167 177770 2$; JMP 1$
3941 020400 005202 3$; INC R2
3942 020402 001401 BEQ ,+4
3943 020404 104400 JMP6; HLT
3944
3945 020406 012767 020424 000000 MOV #13,7$
3946 020414 063767 001004 000012 ADD #67,7$ ;SET UP JMP ADDRESS
3947 020422 000402 BR 2$ ;ADD RELOCATION FACTOR
3948 020424 005102 COM R2 ;GO TO JMP 07$ INST
3949 020426 000403 BR 3$ ;COMPLEMENT INDICATOR
3950 020430 000177 000000 2$; JMP 07$ ;GO TO CHECK ROUTINE
3951 020434 000000 ;WORD 0 ;JUMP TO 1$ ABOVE VIA 7$
3952 020436 005202 INC R2 ;CONTAINS JMP ADDRESS
3953 020440 001401 BEQ ,+4 ;CHECK INDICATOR
3954 020442 104400 JMP7; HLT
3955 020444 104000 SCOPE
3956
3957 ICHECK JSR INSTRUCTIONS
3958 020446 013705 001004 JSRST1; MOV #7,7$ ;GET RELOCATION FACTOR
3959 020452 012702 020504 MOV #33,R2 ;FORM DEST ADDR
3960 020456 060502 ADD R5,R2 ;ADD RELOCATION FACTOR
3961 020460 000277 SCC ;PRESET CC'S
3962 020462 000242 CLV
3963 020464 055142 1$; JSR R5,(R2) ;GO TO 3$ VIA R2
3964 020466 005702 TST R2 ;CHECK INDICATOR
3965 020470 001017 BNE JSR3 ;R2 SHOULD=0
3966 020472 023705 701024 CMP #7,7$ ;CHECK THAT RTS R5 RESTORED R5
3967 020476 001014 BNE JSR1
3968 020502 000414 RR JSR1A
3969 020502 000205 2$; RTS R5 ;RETURN FROM SUBROUTINE

```

DC0KCD 11/40-11/45 CPU EXERCISER MACY11 27(655) 4-SEP-74 1153 PAGE 81
DC0KCD START OF SECTION 3

```

3970 020504 113011 3S; BCC JSR1 ;CHECK THAT JSR DID NOT
3971 020506 124100 BVS JSR1
3972 020510 010007 BNE JSR1 ;AFFECT CC'S
3973 020512 100006 BPL JSR1
3974 020514 005022 CLR R2 ;CLEAR INDICATOR
3975 020516 012704 020466 MOV #1,R4 ;GET UNRELOCATED RETURN ADDRESS
3976 020522 061604 ADD (SP),R4 ;ADD RELOCATION FACTOR (OLD R5)
3977 020524 020405 CMP R4,R5 ;CHECK THAT OLD R5 WAS PLACED ON THE
3978 020526 001765 BEQ 2S ;STACK, & THAT NEW R5 CONTAINS RETURN PC
3979 020530 104400 JSR1; HLT ;ERROR! ABOVE
3980
3981 020532 013704 001004 JSR1A; MOV ##FACTOR,R4 ;GET RELOCATION FACTOR
3982 020536 005000 CLR R0 ;SET INDICATOR
3983 020540 012705 020560 MOV #1,R5
3984 020544 060405 ADD R4,R5 ;SET UP JSR DEFERRED ADRS
3985 020546 010502 MOV R5,R2
3986 020550 012715 020576 MOV #5,(R5)
3987 020554 060415 ADD R4,(R5) ;(R5)=DEST ADRS
3988 020556 000401 BR 2S ;RESERVE WORD FOR ADDRESS
3989 020560 000000 1S; *WORD 0 ;CONTAINS DEST ADRS FOR JSR
3990 020562 004435 2S; JSR R4,(R5); ;JSR TO S5 VIA 1S ABOVE
3991 020564 005200 3S; INC R0 ;CHECK INDICATOR
3992 020566 001013 BNE JSR3
3993 020570 000413 BR JSR3A
3994 020572 005100 4S; COM R0 ;COMPLEMENT INDICATOR
3995 020574 000204 RTS 4 ;RETURN FROM SUBROUTINE
3996 020576 012703 020564 5S; MOV #3,R3 ;GET UNRELOCATED RETURN ADDRESS
3997 020602 061603 ADD (SP),R3 ;ADD RELOCATION FACTOR (OLD R4)
3998 020604 020403 CMP R4,R3
3999 020606 001003 RNE JSR3
4000 020610 005722 TST (R2);
4001 020612 020205 CMP R2,R5 ;CHECK AUTO-INC R5
4002 020614 001766 BEQ 4S ;GO TO RTS
4003 020616 104400 JSR3; HLT ;ERROR ABOVE
4004
4005 020620 013704 001004 JSR3A; MOV ##FACTOR,R4
4006 020624 012405 MOV R4,R5
4007 020626 010703 MOV PC,R3
4008 020630 000405 BR 2S
4009 020632 000405 1S; BR 4S
4010 020634 022323 2S; CMP (R3);,(R3);
4011 020636 000277 SCC
4012 020640 004443 JSR R4;=(R3); ;GO TO 2S
4013 020642 104400 HLT
4014 020644 000414 BR JSR4A
4015 020646 103012 4S; BCC JSR4
4016 020650 102011 BVC JSR4
4017 020652 001010 BNE JSR4
4018 020654 100007 BPL JSR4
4019 020656 012702 020642 MOV #3,R2 ;GET UNRELOCATED RETURN ADDRESS
4020 020662 061602 ADD (SP),R2 ;ADD RELOCATION FACTOR (OLD R4)
4021 020664 020204 CMP R2,R4 ;CHECK THAT CALCULATED RETURN
4022 020666 001002 BNE JSR4 ;PC = NEW R4
4023 020670 005724 TST (R4);

```

DC0KCD 11/40-11/45 CPU EXERCISER MACY11 27(655) 4-SEP-74 1153 PAGE 82
DC0KCD START OF SECTION 3

```

4024 020672 000204 RTS R4
4025 020674 104400 JSR4; HLT
4026
4027
4028 020676 000401 JSR4A; BR 2S
4029 020700 000405 1S; BR 3S
4030 020702 010700 2S; MOV PC,R0
4031 020704 004767 177770 JSR PC,1S
4032 020710 100407 BMI JSR6A
4033 020712 104400 HLT
4034 020714 022020 3S; CMP (R0);,(R0);
4035 020716 020010 CMP R0,(SP) ;CHECK THAT RETURN ADDRESS IS ON THE
4036 020720 001401 BEQ 1,4 ;STACK
4037 020722 104400 HLT
4038 020724 000270 SEN ;SET N
4039 020726 000207 RTS PC
4040 020730 104000 JSR6A; SC0PE
4041
4042 ;CHECK IOT TRAP (AND R0LB/ASLB)
4043 ;THIS TEST CHECKS THAT THE PSW IS CORRECT AFTER THE IOT AND THAT THE
4044 ;NEW PSW (FROM IOTVEC+2) IS CORRECT,
4045 020732 012705 000200 IOTTSY; MOV #IOTVEC,R5 ;SET R5=ADDRESS OF IOTVECTOR
4046 020736 010746 MOV PC;=(R5)
4047 020740 062716 000040 ADD #1,=(SP)
4048 020744 012625 MOV (SP);,(R5); ;LOAD IOT TRAP VECTOR
4049 020746 005020 CLR R0
4050 020750 052740 000200 BIS #PRTY4;,(R0) ;SET PRIORITY LEVEL 4 IN PSW
4051 ;PSW=X XXX X00 001 1X1 000
4052 020754 011015 MOV (R0);,(R5) ;SET IOTVEC+2=PSW ABOVE
4053 020756 011504 MOV (R5);,R4 ;SAVE IN R4
4054 020760 042710 000357 BIC #PRTY+17;,(R0)
4055 020764 052710 000144 BIS #PRTY3+2;,(R0) ;PSW=X XXX X00 001 1X1 000
4056 020770 012003 MOV (R0);,R3 ;R3 = PSW ABOVE
4057 020772 010340 MOV R3;,(R0)
4058 020774 000004 IOT
4059 020776 104400 10S; HLT ;ERROR! IOT FAILED TO TRAP
4060
4061 021000 012002 1S; MOV (R0);,R2 ;GET PSW AFTER IOT TRAP
4062 ;NOTE! R0=0
4063 021002 012715 000200 MOV #PRTY4;,(R5) ;RESTORE IOTVEC+2
4064 021006 012745 002564 MOV #,TYPE;,(R5) ;AND IOTVEC
4065 021012 010746 MOV PC;=(SP) ;FORM PC OF 10S ABOVE
4066 021014 062716 177762 ADD #10;,(SP)
4067 021020 022626 CMP (SP);,(SP); ;CHECK RETURN PC ON STACK
4068 021022 001036 RNE 99S
4069 021024 022603 CMP (SP);,R3 ;CHECK SAVED PSW
4070 021026 001034 BNE 99S
4071 021030 032703 140000 RIT #UM,R3 ;BRANCH TO 3S IF IN USER MODE
4072 021034 100413 BMI 3S
4073 021036 001003 BNE 2S ;BRANCH TO 2S IF IN SUPER MODE
4074 021040 020204 CMP R2,R4 ;CHECK PSW AFTER IOT
4075 021042 001026 RNE 99S
4076 021044 000413 BR 4S
4077

```

```

4078 021046 042704 000000 25I R1C #PUN,R4 ;CLEAR PREV MODE BITS
4079 021052 052704 010000 B1S #PSW,R4 ;SET PREV SUPER MODE
4080 021056 020204 CMP R2,R4 ;CHECK PSW AFTER IOT
4081 021060 001017 RNE 995
4082 021062 000404 BR 45
4083
4084 021064 052704 030000 35I B1S #PUN,R4 ;SET PREV USER MODE
4085 021070 020204 CMP R2,R4 ;CHECK PSW AFTER IOT
4086 021072 001012 RNE 995
4087
4088 021074 005002 45I CLR R2
4089 021076 000261 SEC R0 ;ROTATE R0
4090 021080 106100 ROLB R0 ;UNTIL V SETS (R0=200)
4091 021082 102376 BVC ,+2
4092
4093 021084 106300 ASLB R0 ;SHIFT SHOULD SET CARRY
4094 021086 103004 BCC 995
4095 021100 102003 BVC 995
4096 021112 001002 RNE 995
4097 021114 005700 YST R0
4098 021116 001401 BEQ ,+4
4099 021120 104400 995I HLT ;ERROR! ROL/ASL FAILED TO SET
;CC'S PROPERLY (IF R2=0) OR IN-
;CORRECT PSW AFTER IOT (IF R2 NOT 0)
4100
4101
4102 021122 042704 000340 R1C #PRTY7,R4
4103 021126 010437 177776 MOV R4,#PPSW ;RESTORE PSW
4104 021132 012706 000500 MOV #STKPTR,SP ;RESTORE STACK PTR
4105 021136 104000 SCOPE
4106
4107 ;CHECK EMT TRAP SEQUENCE
4108 021140 005000 CLR R0
4109 021142 010746 MOV PC,=(SP)
4110 021144 062710 ADD #EMT1,,(SP)
4111 021150 012637 000030 MOV (SP)+,#EMTVEC
4112 021154 000262 SEV ;SET V
4113 021156 013737 177776 000032 MOV #PPSW,#EMTVEC+2 ;RETAIN CURRENT PSW ON TRAP
4114 021164 000265 +SEI:SEC
4115 021166 104000 EMT
4116 021170 001433 BEQ EMT1C ;TRAP TO EMT1
4117 021172 104400 HLT ;GO TO EMT1C
4118 021174 102027 EMT1I BVC EMT1B ;ERROR! INCORRECT CC'S WERE SET ON RETURN
;IV! SHOULD BE SET ON EMT TRAP
4119 021176 105100 COMB R0 ;R0=000377,CC'S=1001
4120 021200 105500 ADCB R0 ;R0=000000,CC'S=0101
4121 021202 106000 RORB R0 ;R0=000200,CC'S=1010
4122 021204 102023 BVC EMT1B
4123 021206 100022 BPL EMT1B
4124 021210 000257 CCC
4125 021212 105400 NEGB R0 ;R0=000200,CC'S=1010
4126 021214 102017 BVC EMT1B
4127 021216 100016 BPL EMT1B
4128 021220 000242 CLV ;CLEAR IV!
4129 021222 000261 SEC ;AND SET IC!
4130 021224 105300 ODOB R0 ;R0=000177,CC'S=0011
4131 021226 102012 BVC EMT1B
    
```

```

4132 021230 100411 BHI EMT1B
4133 021232 000242 CLV ;CLEAR IV!
4134 021234 105200 INDB R0 ;R0=000200,CC'S=1011
4135 021236 103006 RCC EMT1B
4136 021240 102005 BVC EMT1B
4137 021242 100004 BPL EMT1B
4138 021244 000242 CLV ;CLEAR IV!
4139 021246 106200 ASMB R0 ;SHIFT R0 UNTIL IV! CLEARS
4140 021250 102776 RNB ,+2
4141 021252 000401 RRB ,+4
4142 021254 104400 EMT1B I HLT ;ERROR!
4143 021256 000002 EMT1C RTI ;EXIT WITH R0=000377
4144 021260 105500 EMT1C ADCB R0 ;R0=000000
4145 021262 103003 BCC EMT1D
4146 021264 001002 RNE EMT1D
4147 021266 005700 TST R0
4148 021270 001401 BEQ ,+4
4149 021272 104400 EMT1D I HLT
4150 021274 012737 001014 000030 MOV #SCOPEA,#EMTVEC ;RESTORE EMT TO SCOPE
4151 021302 005037 000032 CLR #EMTVEC+2
4152 021306 104000 SCOPE
4153
4154 ;CHECK TRAP INSTRUCTION TRAP SEQUENCE
4155 HLT=EMT
4156 021310 013737 000034 000030 MOV #TRAPVEC,#EMTVEC ;REDEFINE HLT
;SET EMT (HLT) TRAP VECTOR
4157 021316 010746 MOV PC,=(SP)
4158 021320 062710 000042 ADD #TRAP1,,(SP)
4159 021324 012637 000034 MOV (SP)+,#TRAPVEC
4160 021330 000270 SEN
4161 021332 013737 177776 000036 MOV #PPSW,#TRAPVEC+2 ;SET N
;RETAIN CURRENT PSW ON TRAP
4162 021340 000261 SEC ;SET CARRY
4163 021342 010700 MOV PC,R0
4164 021344 000264 SEB ;SET Z BIT
4165 021346 104400 TRAP TRAP1 ;TRAP TO TRAP1
4166 021350 103401 BCS ,+4
4167 021352 104000 HLT
4168 021354 001401 BEQ ,+4
4169 021356 104000 HLT
4170 021360 000412 BR TRAP1C
4171 021362 100401 TRAP1I BHI ,+4 ;IN BIT NOT SET ON TRAP
4172 021364 104000 HLT
4173 021366 062700 000004 ADD #4,R0
4174 021372 020016 CMP R0,(SP) ;CHECK LOW BYTE OF RETURN PC ON
;STACK
4175 021374 001401 BEQ ,+4
4176 021376 104000 HLT
4177 021400 124646 CMPB =(SP),=(SP)
4178 021402 032626 RIT =(SP)+,(SP)+ ;RETURN TO INST FOLLOWING TRAP (13)
4179 021404 000002 RTI
4180
4181 021406 012702 000036 TRAP1C I MOV #TRAPVEC+2,R2 ;RESTORE VECTORS
4182 021412 012712 000340 MOV #PRTY7,(R2)
4183 021416 012742 003212 MOV #,HLT,-(R2)
4184 021422 005042 CLR =(R2)
4185 021424 012742 001214 MOV #SCOPEA,-(R2)
    
```

```

4186 02143g 104000 SCOPE
4187 104400 HLT=TRAP ;RESTORE HLT TO A TRAP INST
4188
4189 021432 010702 MOV PC,R2
4190 021434 062702 000012 ADD #12,R2
4191 021440 012707 001152 MOV #RELOC,PC ;GO RELOCATE PROGRAM CODE
4192 021444 000000 REL331 WORD 0
4193 ;333333333333 LAST ADDRESS OF CODE TO BE RELOCATED 3333333333
4194
4195 021446 010701 MOV PC,R1 ;SET SCOPE POINTER
4196 021450 122737 000004 000764 CMPB #4,0#OPT,CP ;BRANCH IF 11/40 OR 11/45
4197 021456 101405 BLOS REL6
4198 021460 012737 000002 031204 MOV #RTI,0#RTI1 ;SET 'I' TRAP RETURN TO RTI
4199 021466 000137 027640 JMP 0#TTYCHK ;JUMP IF 11/05 OR 11/20
4200
4201 ;BRTTL START OF SECTION 4
4202 ;4444444444444444 FIRST ADDRESS TO BE RELOCATED 4444444444
4203 021472 010700 REL41 MOV PC,R0 ;GET PC
4204 021474 005740 TST -(R0) ;R0 CONTAINS THE ADDRESS OF REL4
4205 021476 010037 001010 MOV R0,0#FRSTAD ;SAVE
4206 021502 012737 000004 005176 MOV #4,0#SECT ;SET SECTION #
4207 021510 004737 005166 JSR PC,0#LDDISP ;LOAD DISPLAY GEG
4208 021514 013767 005172 001370 MOV 0#DISPLY,REL44
4209 021522 010700 MOV PC,R0 ;GET CURRENT PC
4210 021524 162700 021524 SUB #,R0 ;SUBTRACT RELOCATION FACTOR
4211 021530 010037 001004 MOV R0,0#FACTOR ;SAVE RELOCATION FACTOR
4212 021534 010701 MOV PC,R1 ;SET NEW SCOPE PTR
4213
4214 ;CHECK STACK OVERFLOW
4215 021536 013767 177776 000332 OVFLW1 MOV 0#PSW,75 ;SAVE STATUS IN 75 BELOW
4216 021544 005037 177776 CLR 0#PSW ;SET KERNEL MODE
4217 021550 004737 002532 JSR PC,0#CLRTRBIT ;GO CLEAR 'I' BIT IF SET
4218 021554 052737 000340 177776 BIS #RTY7,0#PSW ;SET PRIORITY LEVEL 7 TO BLOCK CLOCK
4219 021562 010746 MOV PC,(SP) ;PUSH CURRENT PC ONTO STACK
4220 021564 062716 000146 ADD #25,-1(SP) ;FORM ADDRESS OF 25 BELOW
4221 021570 011637 000004 MOV (SP),0#ERRVEC ;SET ERROR VECTOR
4222 021574 012737 000340 000006 MOV #340,0#ERRVEC*2 ;SET PRIORITY LEVEL 7 ON TRAP
4223 021602 013727 000016 MOV 0#BPTVEC*2,(PC)* ;SAVE CONTENTS OF BPT VECTOR *2
4224 021606 000000 ;WORD 0
4225 021610 062716 000100 ADD #415*25,(SP) ;FORM ADDRESS OF 415 BELOW
4226 021614 012637 000014 MOV (SP)*,0#BPTVEC ;SET BPT TRAP VECTOR TO 415
4227 021620 012737 000340 000016 MOV #340,0#BPTVEC*2
4228
4229 021626 012703 000376 MOV #376,R3
4230 021632 010313 MOV R3,(R3) ;LOAD 376 INTO ADDRESS 376
4231 021634 010306 MOV R3,SP ;SET STACK PTR AT BOUNDARY
4232 021636 032767 140020 000232 RIT #UM,75 ;CHECK IF ENTERED TEST IN KERNEL
4233 021644 001015 BNE 15 ;MODE, BRANCH IF NOT IN KERNEL
4234
4235 ;THE BELOW INSTRUCTIONS SHOULD NOT CAUSE AN OVERFLOW TRAP
4236 021646 005716 177776 TST (SP) ;BECAUSE TST IS A NON MODIFYING INST
4237 021650 021666 177776 CMP (SP)*,-2(SP) ;ISD IS COMPARE
4238 021654 012656 MOV (SP)*,0*(SP) ;BECAUSE OF ADDRESS MODE 5
4239 021656 057636 000020 BIS 0*(SR),0*(SP)+ ;BECAUSE OF ADDRESS MODE 3

```

```

4240 021662 054676 000000 BIS -(SR),0*(SP) ;BECAUSE OF ADDRESS MODE 7
4241 021666 005006 000000 CLR SP
4242 021670 013766 020000 020000 MOV 0#20000,20000(SP)
4243 021676 000473 RR 35 ;BRANCH OVER NON KERNEL MODE TESTS
4244
4245 ;NOTE1 NO OVERFLOW TRAP WILL OCCUR IF NOT IN KERNEL MODE!!!
4246 021700 156737 000173 177777 151 BISB 75*1,0#PSW*1 ;RESTORE MODE BITS IN PSW
4247 021706 012706 000376 MOV #376,SP ;SET STACK PTR
4248 021712 016646 177776 MOV -2(SP),0*(SP) ;SHOULD NOT TRAP
4249 021716 051616 BIS (SP),0*(SP)
4250 021720 061666 177776 ADD (SP)*,-2(SP)
4251 021724 105037 177777 CLRBB 0#PSW*1 ;SET KERNEL MODE
4252 021730 000491 RR 65 ;EXIT TEST
4253
4254 ;ERROR SERVICE ROUTINE
4255 021732 012600 251 MOV (SP)*,R0 ;SAVE PC OF INSTRUCTION THAT TRAPPED
4256 021734 012602 MOV (SP)*,R2 ;SAVE PSW
4257 021736 012706 000500 MOV #STKPTR,SP ;SET STACK PTR
4258 021742 104400 HLT ;ERROR! AN INSTRUCTION THAT WAS NOT
4259 ;SUSPOSED TO TRAP TRAPPED
4260 ;R0 CONTAINS PC, R2 CONTAINS PSW
4261 021744 000443 RR 65 ;EXIT TEST
4262
4263 ;THE BELOW INSTRUCTIONS WILL CAUSE A STACK OVERFLOW
4264 021746 062737 000066 000004 351 ;STACK PTR IS AT 376
4265 021754 010306 ADD #45*25,0#ERRVEC ;SET ERROR VECTOR TO 45
4266 021756 112702 000001 MOV R3,SP ;SET STACK PTR AT 376
4267 021762 005000 MOVBB #1,R2
4268 021764 005016 CLR R0
4269 021766 006302 CLR (SP) ;SETS BIT 0 IN R0
4270 021770 105226 ASL R2 ;SHIFT INDICATOR BIT
4271 021772 006302 INCB (SP)* ;SETS BIT 1 IN R0
4272 021774 060746 ASL R2 ;SETS BIT 2 IN R0
4273 021776 006302 ASL R2
4274 022000 000003 RPT ;SETS BIT 3 IN R0
4275 022002 006302 ASL R2
4276 022004 004767 000014 JSR PC,405 ;SETS BIT 4 IN R0
4277 022010 006302 ASL R2
4278 022012 050666 177776 BIS SP,-2(SP) ;SETS BIT 5 IN R0
4279 022016 000410 RR 55
4280
4281 ;PROGRAM WILL TRAP HERE ON OVERFLOW TRAP
4282 022020 050200 451 BIS R2,R0 ;SET APPROPRIATE BIT IN R0
4283 022022 000002 RTI ;RETURN FROM TRAP
4284
4285 022024 052700 001000 4051 RIS #1000,R0 ;SET IND THAT JSR WAS EXECUTED
4286 022030 000207 RTS PC
4287
4288 022032 052700 000400 4151 RIS #400,R0 ;SET IND THAT BPT WAS EXECUTED
4289 022036 000002 RTI
4290
4291 ;CHECK THAT ABOVE INSTRUCTIONS DID TRAP
4292 022040 012706 000500 551 MOV #STKPTR,SP ;SET STACK PTR
4293 022044 022700 001477 CMP #1477,R0 ;EACH INSTRUCTION SET A BIT IN R0

```

```

4294 022050 001401      BEQ    ,+4          ;R0# 1477
4295 022052 104400      HLT
4296
4297
4298 022054 012706 000600      JEXIT ROUTINE
4299 022060 012737 000016 000014 6S1    MOV    #KPTR,SP      ;SET KERNEL STACK PTR
4300 022066 016737 177514 000016    MOV    #BPTVEC+2,#BPTVEC
4301 022074 012746      MOV    (PC)+,-(SP)   ;PUSH OLD PSW ONTO STACK
4302 022076 000000      WORD   0             ;CONTAINS SAVED PSW
4303 022100 010746      MOV    PC,=(SP)      ;PUSH CURRENT PC ONTO STACK
4304 022102 062716 000006      ADD    #6,(SP)       ;ADD OFFSET
4305 022106 000002      RTI
4306 022110 012706 000500      MOV    #STKPTR,SP    ;SET STACK PTR
4307 022114 012737 000274 000004    MOV    #ERRPT,#ERRVEC ;REBET TIME OUT VECTOR
4308 022122 012737 000002 000006    MOV    #RTI,#ERRVEC+2
4309 022130 104000      SCOPE
4310
4311
4312 022132 012702 022236      ;CHECK THAT ALL RESERVED INSTRUCTIONS TRAP (TO LOCATION 10)
4313 022136 063702 001004      RESTRP1 MOV    #5,R2      ;GET ADDRESS OF RESERVED INSTRUCTION TABLE
4314 022142 132737 000040 000765    ADD    #FACTOR,R2
4315 022150 001402      BITB   #40,#OPT,CP+1 ;CHECK IF 11/45 FLOATING POINT IS AVAIL.
4316 022152 000567 000110      BEQ    ,+6          ;BRANCH IF NOT AVAILABLE
4317 022156 012737 022214 000010    CLR    #5          ;SET TABLE TERMINATOR AT GROUP 7
4318 022164 063737 021024 000010    MOV    #5,#RESVEC   ;SET RESERVED INSTRUCTION TRAP
4319 022172 012203      ADD    #FACTOR,#RESVEC
4320 022174 001437      1S1    MOV    (R2)+,R3      ;GET FIRST RESERVED INSTRUCTION
4321 022176 012204      BEQ    7S          ;IF TERMINATES THE TABLE
4322 022200 010317      MOV    (R2)+,R4      ;GET LAST RESERVED INSTRUCTION IN GROUP
4323 022202 000000      2S1    MOV    R3,(PC)      ;EXECUTE RESERVED INSTRUCTION
4324 022204 104400      3S1    WORD   0             ;CONTAINS RESERVED INSTRUCTION
4325 022206 104400      HLT     ;ERROR! INSTRUCTION IN R3
4326 022210 104400      HLT     ;(2S) ABOVE FAILED TO CAUSE A
4327 022212 000405      BR     ;RESERVED INSTRUCTION TRAP
4328 022214 012716 022226      4S1    MOV    #41S,(SP)    ;ADJUST RETURN PC
4329 022220 063716 021004      ADD    #FACTOR,(SP) ;TO RETURN TO 41S
4330 022224 000002      RTI     ;RETURN TO 41S
4331 022226 020304      41S1  CMP    R3,R4        ;HAS GROUP OF RESERVED INSTRUCTIONS
4332 022230 001760      BEQ    1S          ;BEEN EXECUTED
4333 022232 030203      INC    R3           ;INCREMENT THIS RESERVED INSTRUCTION
4334 022234 000761      BR     2S          ;TO NEXT ONE AND EXECUTE
4335
4336 022236 000007      ;TABLE OF 11/40,11/45 RESERVED INSTRUCTIONS (0 TERMINATES THE TABLE)
4337 022240 000077      5S1    7       ;GROUP 1
4338 022242 000210      77      ;"
4339 022244 000227      210     ;GROUP 2
4340 022246 000227      227     ;"
4341 022248 000227      7000    ;GROUP 3
4342 022250 000227      7777    ;"
4343 022252 075040      75040   ;GROUP 4
4344 022254 076977      76977   ;"
4345 022256 106400      106400  ;GROUP 5
4346 022258 106477      106477  ;"
4347 022260 106700      106700  ;GROUP 6
4348 022262 107777      107777  ;"

```

```

4348 022266 170000      50S1  170000      ;GROUP 7      FLOATING POINT
4349 022270 177777      177777      ;"          INSTRUCTIONS
4350 022272 000000      0           ;B TERMINATES THE TABLE
4351
4352 022274 012737 000264 000010 7S1    MOV    #RESERR,#RESVEC ;RESTORE RESERVED TRAP
4353 022302 104000      SCOPE
4354
4355
4356 022304 105737 000770      ;CHECK THAT ALL BITS IN THE PROCESSOR STATUS WORD (PSW) CAN BE SET AND
4357 022310 001072      CLEARED.
4358 022312 013767 177776 000144 000CHK1 TSTB   #MMON      ;IF MEM MGMT IS ON SKIP THIS TEST
4359 022314 013767 177776      BNC    4S          ;SAVE STATUS
4360 022316 000037 177776      MOV    #PSW,3S     ;CLEAR MODE BITS IN PSW
4361 022320 000037 002532      CLR    #PSW        ;GO CLEAR '1' BIT IF SET
4362 022324 004737 000016      JSR    PC,#CLR1BIT
4363 022330 013746 000016      MOV    #BPTVEC+2,#(SP) ;LOAD ADDRESS OF PSW INTO R4
4364 022334 012704 177776      MOV    #PSW,R4
4365 022336 000230      CLN
4366 022340 000230      TST   (R4)         ;CHECK THAT PSW WAS CLEARED
4367 022342 000714      BEQ    ,+4          ;ERROR! PSW FAILED TO CLEAR
4368 022344 104400      HLT
4369 022346 104400      MOVB   #00BT,CP,R0 ;GET CP TYPE
4370 022348 113700 000764      MOV    PSWBIT(0),R0 ;GET BIT MASK FOR TEST 0=THOSE BITS IN
4371 022350 010000      HLT     ;THE PSW WHICH CAN BE SET/CLEARED,
4372 022352 012702 000001      TST   #00RT,CP    ;CHECK IF MEM MGMT IS AVAILABLE
4373 022354 052700 170000      BPL   10S         ;BRANCH IF NOT AVAILABLE
4374 022356 012702 000001      10S1  MOV    #170000,R0  ;SET BITS 10-12 IF MEM MGMT
4375 022358 030200      1S1    BIT   R2,R0        ;R0 = TEST BIT
4376 022360 001403      BEQ    2S          ;CHECK IF BIT CAN BE SET/CLEARED
4377 022362 000003      CLR    #BTIVEC+P
4378 022364 030227 000020      BIT   R2,#20      ;CHECK IF TEST WILL SET '1' BIT
4379 022366 001403      BEQ    20S         ;RTI,#BTIVEC+2; SET RTI INTO RETURN
4380 022368 012737 000002 000016 20S1  CLR    (R4)        ;CLEAR PSW
4381 022370 000014      BIC   R2,(R4)     ;SET R0 INTO PSW
4382 022372 050214      MOV    R2,(R3)    ;GET BIT
4383 022374 011403      CMP    R2,R3      ;CHECK THAT BIT WAS SET IN PSW
4384 022376 000002      BEQ    ,+4
4385 022378 001403      HLT     ;ERROR! BIT IN R2 FAILED TO SET IN PSW
4386 022380 104400      CLR    2          ;CLEAR 2 BIT
4387 022382 000244      BIC   R2,(R4)    ;CLEAR BIT IN PSW
4388 022384 040214      MOV    (R4),R3   ;GET PSW RESULT
4389 022386 011403      BEQ    2S        ;BRANCH IF BIC ABOVE CLEARED BIT IN PSW
4390 022388 001403      HLT     ;ERROR! BIT IN R2 FAILED TO CLEAR IN PSW
4391 022390 104400      2S1  ASL   R2         ;SHIFT TEST BIT
4392 022392 000302      RCC   1S          ;BRANCH IF ALL BITS NOT TESTED
4393 022394 103351      CLR    (R4)      ;CLEAR STATUS
4394 022396 000014      MOV    (SP)+,#BTIVEC+2 ;RESTORE 1 BIT RETURN
4395 022398 012637 000010      MOV    (PC)+,#(SP)  ;PUSH ORIGINAL STATUS ON STACK
4396 022400 012746 000000      WORD   0             ;CONTAINS ORIGINAL PSW
4397 022402 000000      MOV    PC,=(SP)   ;SET RETURN PC
4398 022404 010746      ADD    #6,(SP)
4399 022406 062716 000206      RTI
4400 022408 000002      SCOPE
4401 022410 104000

```

```

4402
4403 022500 013704 177776      MOV      #0PSW,R4      ;SAVE PSW IN R4
4404 022504 112737 000300 177776  MOVB    #300,#PSW    ;SET PRIORITY LEVEL 6
4405 022512 004737 002532      JSR      PC,#CLRTBIT ;"CLEAR IT" BIT IF SET
4406
4407      ;CHECK THAT ALL BITS IN THE CURRENT STACK PTR CAN BE SET/CLEARED
4408 022516 010603      CHKSP;  MOV      SP,R3      ;SAVE STACK PTR
4409 022520 000257      CCC
4410 022522 112706 000377      MOVB    #377,SP      ;SET STACK PTR = 377
4411 022526 006006 15;  ROR      SP          ;ROTATE 0 BIT THROUGH ALL BIT
4412 022530 103776      BCS     1$           ;BIT POSITIONS
4413 022532 005206      INC     SP          ;SHOULD INCREMENT SP TO 0
4414 022534 001403      BEQ     2$
4415 022536 010602      MOV     SP,R2      ;SAVE ERROR STACK PTR
4416 022540 010306      MOV     R3,SP      ;SET STACK PTR FOR TRAP
4417 022542 104400      HLT
4418
4419 022544 010306 25;  MOV     R3,SP      ;RESTORE ORIGINAL STACK PTR
4420
4421      ;CHECK BYTE OPERATIONS USING THE STACK
4422 022546 010600      SPCMK;  MOV     SP,R0      ;SAVE STACK PTR
4423 022550 010003      MOV     R0,R3
4424
4425 022552 005043      CLR     -(R3)
4426 022554 112746 177777      MOVB    #1,*(SP)    ;(SP) = 377
4427 022560 022713 000377      CMP     #377,(R3)   ;CHECK THAT ONLY EVEN BYTE WAS AFFECTED
4428 022564 001002      BNE     1$
4429 022566 020306      CMP     R3,SP      ;CHECK AUTO-DEC
4430 022570 001401      BEQ     ,+4
4431 022572 104400 15;  HLT
4432
4433 022574 105226      INCB   (SP)+
4434 022576 005723      TST    (R3)+      ;CHECK RESULT
4435 022600 001002      BNE     2$
4436 022602 020006      CMP     R0,SP      ;CHECK AUTO-INC
4437 022604 001401      BEQ     ,+4
4438 022606 104400 25;  HLT
4439
4440 022610 005143      COM    -(R3)      ;(R3)=17777
4441 022612 144613      RLCB   -(SP),(R3)
4442 022614 022713 177400      CMP     #177400,(R3) ;CHECK RESULT
4443 022620 001002      BNE     3$
4444 022622 020603      CMP     SP,R3
4445 022624 001401      BEQ     ,+4
4446 022626 104400 35;  HLT
4447
4448 022630 132627 000377      BITB   (SP)+,#377
4449 022634 001002      BNE     4$
4450 022636 020600      CMP     SP,R0
4451 022640 001401      BEQ     ,+4
4452 022642 104400 45;  HLT
4453
4454 022644 012746 000001      MOV     #1,(SP)
4455 022650 062706 000002      ADD     #2,SP

```

```

4456 022654 012702 177401      MOV     #177401,R2
4457 022660 120246      CHPB   R2,*(SP)
4458 022662 001004      BNE     5$
4459 022664 122602      CMPB   (SP)+,R2
4460 022666 001002      BNE     5$
4461 022670 020006      CMP     R0,SP
4462 022672 001401      BEQ     ,+4
4463 022674 104400 55;  HLT
4464 022676 105437 177776      CLRB   #PSW
4465 022702 010446      MOV     R4,*(SP)    ;RESTORE ORIGINAL PSW TO STACK
4466 022704 010746      MOV     PC,#PSW
4467 022706 062716 000006      ADD     #6,4SP)
4468 022712 000002      RTI
4469 022714 104400      SCOPE
4470
4471      ;CHECK THAT 'C' BIT SETS/CLEAR PROPERLY
4472 022716 012727 177776      CBITI  MOV     #177776,(PC)+ ;LOAD CONSTANT
4473 022722 000000 15;  WORD    0
4474 022724 010700      MOV     PC,R0      ;GET CURRENT PC
4475 022726 162700 000004      SUB     #4,R0      ;POINT R0 TO 1$ ABOVE
4476 022732 005520 25;  ADC     (R0)+      ;ADD 'C' BIT TO 1$ ABOVE
4477 022734 006340      ASL    -(R0)
4478 022736 102375      BVC    2$
4479 022740 022767 077776 177754      CMP     #077776,1$ ;CHECK RESULT
4480 022746 001401      BEQ     ,+4
4481 022750 104400      HLT
4482
4483      ;CHECK THAT CONDITION CODES ARE SET PROPERLY WHEN A NUMBER (CURRENT PC)
4484      ;AND THAT NUMBER +1 ARE COMPARED, AND VICE VERSA,
4485      CHPN;  MOV     PC,R0      ;GET CURRENT PC
4486 022752 010700      MOV     R0,R2      ;SAVE IN R2
4487 022754 010002      INC     R2          ;MAKE R2 = R0+1
4488 022756 005202      SCC
4489 022760 000277      +CLC;  CLN
4490 022762 000251      CMP     R0,R2      ;CLEAR C & N BITS
4491 022764 020002      BCC    1$          ;COMPARE # WITH #+1
4492 022766 103003      BVS    1$          ;CARRY BIT SHOULD SET
4493 022770 102402      BVS    1$          ;V BIT SHOULD CLEAR
4494 022772 001401      BEQ    1$          ;Z BIT SHOULD CLEAR
4495 022774 100401 15;  BMI    ,+4        ;N BIT SHOULD SET
4496 022776 104400      HLT
4497
4498      ;ERROR! INCORRECT RESULT IN 1$ ABOVE
4499      ;R0=ADDRESS OF DATA
4500      ;CHECK THAT CONDITION CODES IN PSW CORRECTLY
4501      SCC
4502      CMPB   R2,R0      ;COMPARE #+1 WITH #
4503      BCS     2$          ;C BIT SHOULD CLEAR
4504      BVS    2$          ;V BIT SHOULD CLEAR
4505      BEQ    2$          ;Z BIT SHOULD CLEAR
4506      BPL    ,+4        ;N BIT SHOULD CLEAR
4507      HLT
4508      ;ERROR! COMPARE #+1 WITH # FAILED TO SET
4509      ;CONDITION CODES IN PSW CORRECTLY

```


4510
 4511 023016 000240
 4512 023020 000240
 4513 023022 000240
 4514 023024 000240
 4515 023026 000240
 4516 023030 000240
 4517 023032 000240
 4518 023034 000240
 4519 023036 000240
 4520 023040 000240
 4521 023042 000240
 4522 023044 000240
 4523 023046 000240
 4524 023050 000240
 4525 023052 000240
 4526 023054 000240
 4527 023056 000240
 4528 023060 000240
 4529 023062 000240
 4530 023064 000240
 4531 023066 000240
 4532 023070 000240
 4533 023072 000240
 4534 023074 000240
 4535 023076 104000
 4537 023100 010702
 4538 023102 062702 000012
 4539 023106 012707 001152
 4540 023112 000000
 4541
 4542
 4543
 4544
 4545
 4546
 4547 023114 010700
 4548 023116 005740
 4549 023120 010037 001010
 4550 023124 012737 000005 005176
 4551 023132 004737 005166
 4552 023136 013767 001462
 4553 023144 010700
 4554 023146 162700 023146
 4555 023152 010037 001004
 4556 023156 010701
 4557
 4558
 4559 023160 005000
 4560 023162 000277
 4561 023164 000700
 4562 023166 103000
 4563 023172 102404

BE POSITION INDEPENDENT AND SHOULD RUN WHEN RELOCATED BY THE PROGRAM,

NOP
 NOP
 NOP
 NOP
 NOP
 NOP
 NOP
 NOP
 NOP
 NOP
 NOP
 NOP
 NOP
 NOP
 NOP
 NOP
 NOP
 NOP
 NOP
 NOP
 NOP
 NOP
 NOP
 NOP
 NOP
 NOP
 MOV PC,R2
 ADD #12,R2
 MOV #RELOC,PC ;GO RELOCATE PROGRAM CODE
 REL44: .WORD 0
 ;44444444444444 LAST ADDRESS OF CODE TO BE RELOCATED 4444444444

START OF SECTION 5
 5555555555555555 FIRST ADDRESS TO BE RELOCATED 5555555555
 REL5: MOV PC,R0 ;GET PC
 TST +(R0) ;R0 CONTAINS THE ADDRESS OF REL5
 MOV R0,#FRSTAD ;SAVE
 JSR #5,#R0,REL5 ;SET SECTION #
 MOV #0,R0 ;LOAD DISPLAY REG
 MOV PC,R0 ;GET CURRENT PC
 SUB #,R0 ;SUBTRACT RELOCATION FACTOR
 MOV R0,#FACTOR ;SAVE RELOCATION FACTOR
 MOV PC,R1 ;SET NEW SCOPE PTR
 ;CHECK EXTENDED INSTRUCTION SET (SXT, XOR, SOB, MARK, RTI/RTT)
 EXTINSTCLR R0
 SCC ;PRESET CC'S
 SXT R0 ;EXTEND SIGN (1) INTO R0
 BCC SXT0 ;CHECK RESULT CC'S
 BVS SXT0

4564 023172 001403
 4565 023174 100002
 4566 023176 005200
 4567 023200 001401
 4568 023202 104400
 4569
 4570 023204 010700
 4571 023206 010002
 4572 023210 012703 177777
 4573 023214 005102
 4574 023216 000243
 4575 023220 074003
 4576 023222 103404
 4577 023224 102403
 4578 023226 001402
 4579 023230 020203
 4580 023232 001401
 4581 023234 104400
 4582
 4583 023236 010700
 4584 023240 022020
 4585 023242 000401
 4586 023244 000000
 4587 023246 005700
 4588 023250 006710
 4589 023252 005002
 4590 023254 005700
 4591 023256 100001
 4592 023260 005102
 4593 023262 021002
 4594 023264 001401
 4595 023266 104400
 4596
 4597 023270 012710 100000
 4598 023274 011002
 4599 023276 000277
 4600 023300 074210
 4601 023302 103007
 4602 023304 102406
 4603 023306 001005
 4604 023310 100404
 4605 023312 005710
 4606 023314 001002
 4607 023316 005402
 4608 023320 102401
 4609 023322 104400
 4610
 4611 023324 010702
 4612 023326 022222
 4613 023330 000401
 4614 023332 000000
 4615 023334 012722 125252
 4616 023340 006742
 4617 023342 074722

BEO SXT0
 BPL SXT0
 INC R0 ;CHECK RESULT
 BEO SXT0
 HLT
 MOV PC,R0
 MOV R0,R2
 MOV R0,R3
 COM R2
 +CLVCLC
 XOR R0,R3 ;CLEAR C AND V BITS
 XOR R0,R0 ;R3 SHOULD CONTAIN COMPLEMENT OF R0
 BVS X0R0 ;CHECK THAT C WAS NOT AFFECTED
 BEO X0R0 ;AND THAT V WAS CLEARED
 CMP R2,R3 ;CHECK RESULT
 BEO SXT0
 HLT ;ERROR! XOR FAILED
 MOV PC,R0
 CMP (R0),R0 ;SET ADDRESS REGISTER
 BR 15 ;RESERVE WORD FOR TEST DATA
 .WORD 0 ;CONTAINS TEST DATA
 TST R0 ;EXTEND SIGN OF ADDRESS INTO
 SXT (R0) ;ADDRESS (R0)=1 IF MSB R0=1
 CLR R2 ;OTHERWISE, (R0)=0
 TST R0 ;CHECK SIGN OF ADDRESS
 BPL SXT0
 R2 ;COMPLEMENT CHECK REG IF NEG
 COM (R0),R2 ;CHECK RESULT OF SXT
 BEO SXT0
 HLT ;ERROR! SXT FAILED TO EXTEND SIGN PROPERLY
 MOV #100000,(R0) ;PRESET DATA
 MOV (R0),R2
 SCC ;PRESET CC'S
 XOR (R0) ;XOR 100000 WITH 100000 RESULT = 0
 BCC X0R0 ;CHECK CC'S AFTER XOR
 BVS X0R0
 BNE X0R0
 BHI X0R0
 TST (R0) ;CHECK RESULT (0)
 XOR X0R0
 NEG R2 ;CHECK THAT REG WAS NOT AFFECTED
 BVS SXT0
 HLT
 MOV PC,R2
 CMP (R2),R2 ;RESERVE WORD FOR DATA
 BR SXT0 ;RESERVED FOR DATA
 .WORD 0 ;PRESET DATA
 MOV #125052,(R2) ;PRESET DATA
 SXT (R2) ;EXTEND SIGN
 XOR PC,(R2)

```

DC0KCD 11/40-11/44 CPU EXERCISER          MACY11 27(655) 4-SEP-74 1153 PAGE 93
DC0KCD  START OF SECTION 5

4618 223344 010700          MOV      PC,R0          JGET PC
4619 223346 005740          TST     -(R0)         JSUBTRACT 2 FROM PC
4620 223350 005100          COM     R0            IR0=RESULT OF XOR PC-1 ABOVE
4621 223352 074042          XOR     R0,-(R2)     ICHECK RESULT OF SXT AND XOR ABOVE
4622 223354 001401          BEQ     ,+4           J,+4
4623 223356 104400          XOR24; HLT           JERROR! SXT & XOR ABOVE INCORRECT
4624
4625 223360 012704 000001          MOV     #1,R4         JSET R4
4626 223364 006767 000060          SXT    XOR6A        IPRESET DATA=0
4627 223370 074467 000054          2$;   XOR     R4,XOR6A
4628 223374 100423          BMI    XOR6         JXOR6
4629 223376 006304          ASL    R4           ISHIFT R4
4630 223400 102373          BVC   XOR6         JUNTIL V SETS (R4=100000)
4631 223402 100020          BPL   XOR6         JBRANCH IF 'N' IS CLEAR
4632 223404 074467 000040          XOR     R4,XOR6A    IXOR6A=177777
4633 223410 100015          BPL   XOR6
4634 223412 074767 000032          XOR     PC,XOR6A    IXOR PC WITH XOR6A (177777)
4635 223416 010767 000030          MOV     PC,XOR6B    IFORN PC AS USED IN XOR ABOVE
4636 223422 102767 000004          SUB    #4,XOR6B
4637 223430 005167 000016          COM    XOR6B
4638 223434 026767 000012 000006          CMP    XOR6B,XOR6A JXOR6A SHOULD = COMPLEMENT OF PC
4639 223442 001401          BEQ     ,+4
4640 223444 104400          XOR6; HLT           JERROR! XOR TESTS ABOVE FAILED
4641
4642 223446 000402          BR     ,+6
4643
4644 223450 000000          XOR6A; WORD 0       JCONTAINS DATA USED BY TEST ABOVE
4645 223452 000000          XOR6B; WORD 0
4646
4647
4648 223454 012700 077777          MOV     #077777,R0  JSET SOURCE OPERAND FOR ADC
4649 223460 006767 177764          SXT    XOR6A        JCLEAR XOR6A
4650 223464 001004          BNE    SXT6        ICHECK CC'S AFTER EXTENDING ZERO'S
4651 223466 100403          BMI    SXT6
4652 223470 103402          BCS   SXT6
4653 223472 102401          BVS   SXT6
4654 223474 000401          BR     ,+4
4655 223476 104400          SXT6; HLT           JERROR! SXT FAILED
4656
4657 223500 012722 000001          MOV     #1,R2        JSET DEST OPERAND FOR ADD
4658 223504 013703 001004          MOV     ##FACTOR,R3 JLOAD INDEX REGISTER
4659 223510 000002          ADD    R0,R2        JRESULT OF ADD=100000
4660 223512 006763 023450          SXT    XOR6A(3)     JEXTEND SIGN OF ADD ABOVE
4661 223516 001403          BEQ     SXT6A
4662 223520 005267 177724          INC    XOR6A        JCHECK RESULT OF SXT
4663 223524 001401          BEQ     ,+4
4664 223526 104400          SXT6A; HLT          JERROR! SXT ABOVE FAILED TO EXTEND SIGN
4665
4666 223530 010703          MOV     PC,R3
4667 223532 000402          BR     ,+6         JPRESERVE 2 WORDS FOR DATA
4668 223534 000000          SXRA; WORD 0       JRESERVED WORD FOR DATA
4669 223536 000000          SXRB; WORD 0       JRESERVED WORD FOR DATA
4670 223540 005723          TST    (R3)+
4671 223542 010304          MOV     R3,R4        JR3 = ADDRESS OF SXRA

```

```

DC0KCD 11/42-11/45 CPU EXERCISER          MACY11 27(655) 4-SEP-74 1153 PAGE 94
DC0KCD  START OF SECTION 5

4672 223544 000250          CLN    ,+4          JCLEAR N BIT
4673 223546 006724          SXT    (R4)+        JEXTEND ZEROS INTO SXRA
4674 223550 001401          BEQ     ,+4
4675 223552 104400          SXT2; HLT           JERROR! SXT FAILED
4676
4677 223554 010467 177754          MOV     R4,SXRA     JSXRA = ADDRESS OF SXRB
4678 223560 000257          CCC    ,+4          JCLEAR CONDITION CODES
4679 223562 006733          SXT    ,(R3)+      JEXTEND ZEROS INTO SXRB
4680 223564 001401          BEQ     ,+4
4681 223566 104400          SXT3; HLT           JERROR!
4682
4683 223570 000270          SEN    ,+4          JSET N BIT
4684 223572 006753          SXT    0-(R3)     JEXTEND ONES INTO SXRB
4685 223574 100401          BMI    ,+4
4686 223576 104400          SXT5; HLT           JERROR!
4687
4688 223600 012704 025252          MOV     #025252,R4  JR4 = 025252
4689 223604 074433          XOR    R4,(R3)+    JSXRB = 152525 (COMPLEMENT OF R4)
4690 223606 005002          CLR    R2
4691 223610 074253          XOR    R2,(R3)     JSXRB REMAINS UNCHANGED
4692 223612 001405          BEQ    XOR45       ICHECK CONDITION CODES
4693 223614 100004          BPL   XOR35
4694 223616 005104          COM    R4          JR4 = 152525
4695 223620 020467 177712          CMP    R4,SXRB     JCHECK XOR
4696 223624 001401          BEQ    ,+4
4697 223626 104400          XOR35; HLT          JERROR! XOR FAILED
4698
4699 223630 005743          TST    =(R5)       JR3 = ADDRESS OF SXRA-2
4700 223632 000250          CLN    ,+4          JCLEAR N BIT
4701 223634 006773 000002          SXT    #2(R3)     JSXRB = 0
4702 223640 001401          BEQ    ,+4
4703 223642 104400          SXT7; HLT          JERROR! SXT FAILED
4704
4705 223644 074473 000002          XOR    R4,#2(R3)   JSXRB = R4
4706 223650 020473 000002          CMP    R4,#2(R3)   JCHECK XOR
4707 223654 001401          BEQ    ,+4
4708 223656 104400          XOR7; HLT          JERROR! XOR FAILED
4709 223660 104000          SCOPE
4710
4711 ;NOTE; DO NOT INSERT ANY CODE IN FOLLOWING SOB TESTS
4712 ; SINCE IT TESTS THE MAXIMUM BRANCH WIDTH OF THE INSTRUCTION,
4713
4714 223662 005005          CLR    R5          JCLEAR ERROR INDICATOR
4715 223664 000407          BR     SOB0        JBRANCH TO SOB TEST
4716
4717 223666 005004          SOB10; CLR R4       JR4 = 0
4718 223670 005705          TST    R5          JCHECK ERROR INDICATOR
4719 223672 001401          BEQ    ,+4        JSOB BRANCHED CORRECTLY
4720 223674 104400          HLT
4721
4722 223676 005025          SOB9; CLR R5       JCLEAR INDICATOR (R5)
4723 223700 006004          ROR    R4          JROTATE RIGHT R4
4724 223722 000467          BR     SOB8
4725

```

4726 023704 012700 000010
4727 023710 000277
4728 023712 001012
4729 023714 100011
4730 023716 102010
4731 023720 103007
4732 023722 077005
4733 023724 001005
4734 023726 100004
4735 023730 102003
4736 023732 103002
4737 023734 005700
4738 023736 001401
4739 023740 104400
4740
4741 023742 012702 000100
4742 023746 012700 000101
4743 023752 001414
4744 023754 100413
4745 023756 102412
4746 023760 103411
4747 023762 005300
4748 023764 020002
4749 023766 001006
4750 023770 000257
4751 023772 077211
4752 023774 001403
4753 023776 100402
4754 024000 005702
4755 024002 001401
4756 024004 104400
4757
4758 024006 012700 000001
4759 024012 000401
4760 024014 104400
4761 024016 077002
4762
4763 024020 005700
4764 024022 001401
4765 024024 104400
4766
4767 024026 012704 100000
4768 024032 000403
4769 024034 005204
4770 024036 100403
4771 024040 104400
4772
4773
4774 024042 077404
4775 024044 104400
4776
4777 024046 012703 000100
4778 024052 077301
4779 024054 005703

SOB0: MOV #10,R0 ;R0=10
SOB1: SCC ;SET CONDITION CODES
;CHECK CONDITION CODES AFTER SOB
BPL SOB2 ;SOB SHOULD NOT EFFECT THE
;CONDITION CODES,
BVC SOB2
BCC SOB2
SOB R0,SOB1
BNE SOB2 ;CHECK CONDITION CODES AFTER
;SOB FALLS THROUGH,
RPL SOB2 ;SOB SHOULD NOT EFFECT
;CONDITION CODES,
RVC SOB2 ;CHECK IF R0=0
BCC SOB2
TST R0
BEQ ,+4
SOB2: HLT ;ERROR!

MOV #100,R2 ;R2=100
MOV #100,R0 ;SET CHECK REGISTER, R0=100
SOB3: BEO SOB4 ;CHECK CONDITION CODES AFTER
;SOB BRANCH,
;SOB SHOULD NOT EFFECT
;CONDITION CODES,
BCS SOB4 ;DECREMENT CHECK REGISTER
DEC R0 ;CHECK THAT SOB DECREMENTS
CMP R0,R2
BNE SOB4
CCC SOB R2,SOB3 ;SET CONDITION CODES BEFORE SOB
SOB ;BRANCH TO SOB3 UNTIL R2=0
BEO SOB4 ;CHECK CONDITION CODES AFTER
;SOB FALLS THROUGH
;CHECK IF R2=0
R2
SOB4: HLT ;ERROR!

MOV #1,R0 ;R0=1
BR ,+4
HLT ;ERROR!
SOB R0,=2 ;SOB SHOULD NOT BRANCH

TST R0 ;CHECK IF R0=0 AFTER SOB
BEQ ,+4
HLT ;ERROR!

MOV #100000,R4 ;R4=100000
BR 1\$
;S: INC R4 ;R4=100000
;M: ;IN BIT SHOULD BE SET
;E: ;ERROR! SOB DID NOT
;R: ;INCREMENT PROPERLY
HLT

;S: SOB R4,3\$;SOB SHOULD BRANCH
;E: ;ERROR! SOB DID NOT BRANCH
HLT

;S: MOV #100,R3 ;R3=100
;E: SOB R3,SOB6 ;USE SOB TO BRANCH TO ITSELF
;R: TST R3 ;CHECK IF R3=0

4780 024056 001703
4781 024060 104400
4782
4783 024062 005703
4784
4785
4786
4787
4788
4789 024064 001401
4790 024066 104400
4791
4792 024070 005205
4793 024072 077477
4794 024074 005704
4795 024076 001401
4796 024120 104400
4797 024102 104000
4798
4799
4800
4801
4802
4803 024104 010602
4804 024106 010705
4805 024110 010500
4806 024112 010546
4807 024114 010746
4808 024116 010746
4809 024120 010746
4810 024122 010746
4811 024124 010746
4812 024126 012746 006405
4813 024132 010605
4814 024134 004767 002022
4815 024140 000403
4816 024142 000205
4817 024144 104400
4818 024146 000407
4819 024150 020602
4820 024152 001402
4821 024154 104400
4822 024156 000403
4823 024160 020005
4824 024162 001401
4825 024164 104400
4826 024166 010206
4827 024170 104000
4828
4829
4830
4831
4832
4833

SOB7: BEO SOB10
HLT ;ERROR!

SOB8: TST R5 ;CHECK INDICATOR (R5)
;IF SOB BRANCHES INCORRECTLY
;WHEN CHECKING MAX. BRANCH,
;R5 WILL NOT BE CLEARED AT
;THIS POINT INDICATING AN ERROR,

BEQ ,+4 ;BRANCH IF SOB BRANCHES CORRECTLY
HLT ;ERROR!

INC R5 ;SET INDICATOR (R5)
SOB R4,SOB9 ;TEST MAX. BRANCH OF SOB
TST R4 ;CHECK IF R4=0
BEQ ,+4
HLT ;ERROR!
SC0PE

;CHECK THAT MARK INSTRUCTION POPS OVER THE CORRECT # OF ARGUMENTS, RESTORES R5 FROM THE STACK POINTER

MRKTSY: MOV SP,R2
MOV PC,R5 ;THE STACK LOOKS LIKE THIS AFTER
MOV R5,R0 ;THE JSR INSTRUCTION
MOV R5,=(SP) ;-2(SP)=R0 THIS IS A
MOV PC,=(SP) ;-4(SP)=PC STRING
MOV PC,=(SP) ;-6(SP)=PC+2 OF
MOV PC,=(SP) ;-10(SP)=PC+4 FIVE
MOV PC,=(SP) ;-12(SP)=PC+6 DUMMY
MOV PC,=(SP) ;-14(SP)=PC+10 ARGUMENTS
MOV #MARK45,=(SP) ;-16(SP)=MARK 5
MOV SP,R5 ;-20(SP)=PC PUSHED BY JSR
JSR PC,MARK1
BR ,+10
MARK1: RTS R5 ;ERROR! SHOULD BE DOING MARK 5 INST,
BR MARKEX
CMP SP,R2
BEQ ,+6
HLT ;ERROR! SP NOT RETURNED TO PROPER
;VALUE BY MARK INSTRUCTION
BR MARKEX
CMP R0,R5
REQ ,+4
HLT ;ERROR! DID NOT RESTORE R5 FROM STACK
MARKEX: MOV R2,SP ;RESTORE SP
SC0PE

```
4834 ; INC R0 ;R0=000001
4835 ; ; ;IT! TRAP TO 5$ AFTER INC
4836 ;SS: COM R0 ;R0=177776
4837 ; MOV SAVPSW,2(SP) ;CLEAR 'T' BIT IN RETURN PSW
4838 ; RTI ;RETURN TO INSTRUCTION FOLLOWING INC
4839 ; CMP #RTT,2$ ;CHECK
4840 ; ETC
4841 ;
4842 ;INSTRUCTION SEQUENCE=RTI
4843 ;2$ RTI ;IT! TRAP AFTER RTI
4844 ;SS: COM R0 ;R0=177777
4845 ; MOV SAVPSW,2(SP) ;CLEAR 'T' BIT IN RETURN PSW
4846 ; RTI ;RETURN TO INC INSTRUCTION
4847 ; INC R0 ;R0=000000
4848 ; CMP #RTT,2$ ;CHECK
4849 ; ETC
4850 024172 213767 177776 000166 RTT1: MOV 0#PSW,SAVPSW ;SAVE PSW
4851 024200 032767 000020 000160 BIT #T,SAVPSW ;CHECK IF 'T' BIT SET
4852 024206 001176 BNE RTT2EX ;BRANCH TO EXIT
4853 024210 010746 1$: MOV PC,=(SP) ;GET CURRENT PC
4854 024212 062716 000116 ADD #5$,,(SP) ;FORM RELOCATED PC
4855 024216 012637 000014 MOV (SP),#TBITVEC ;LOAD INTO TRAP VECTOR
4856 024222 016746 000140 MOV SAVPSW,#(SP) ;GET CURRENT PSW
```

```
4857 024226 011637 000016 MOV (SP),#TBITVEC*2
4858 024232 052737 000340 177776 BIS #PRTY,0#PSW ;SET PRIORITY LEVEL 7
```

```

DCQKCD 11/40-11/45 CPU EXERCISER MACy11 27(695) 4-SEP-74 11:53 PAGE 99
DCQKCD START OF SECTION 5
4859 024240 005000 CLR R0
4860 024242 052716 000360 BIC #PRTY7+,(SP) ; ISET #TBIT IN PSW ON STACK
4861 024246 010746 MOV PC,=(SP) ; PUT THE PC ON THE STACK
4862 024250 062716 000006 ADD #6,(SP) ; ADJUST PC FOR NEXT INSTRUCTION
4863 024254 000006 RTT
4864 024256 005200 INC R0 ; DONE TO SEE IF INSTR, FOLLOWING
4865 RTT IS EXECUTED IF T-BIT SET
4866 024260 042737 000340 177776 BIC #PRTY7,0#PSW ; ISET PRIORITY LEVEL 0
4867 024266 022767 000006 177760 CMP #RTT,25
4868 024274 001005 BNE 35 ; CHECK IF INC WAS EXECUTED
4869 024276 022700 177776 CMP #177776,R0 ; CHECK IF COM-R0 EXECUTED
4870 024302 001406 BEQ 45
4871 024304 104400 HLT ; ERROR!R0 NOT COMPLIMENTED
4872 024306 000415 BR 65 ; EXIT TEST
4873 024310 005700 35 RTT ; TEST IF TRAPED BEFORE INC INST,
4874 ; WAS EXECUTED
4875 024312 001413 BEQ 65
4876 024314 104400 HLT ; ERROR!
4877 024316 000411 BR 65 ; EXIT TEST
4878 024320 012767 000002 177726 45 MOV #RT1,25
4879 024326 000730 BR 15
4880 024330 005100 55 COM R0 ; RTT CHECK
4881 024332 016766 000030 000002 MOV SAVPSW,2(SP)
4882 024340 000002 RTI
4883 024342 012767 000006 177704 65 MOV #RTT,25
4884 024350 012737 000016 000014 MOV #TBITVEC+2,*TBITVEC ; RESTORE TRAP VECTORES
4885 024356 005037 000016 CLR *TBITVEC+2
4886 RTT1EXI SCOPE
4887 BR RTT2
4888 024364 000401 SAVPSW1,WORD 0
4889 024366 000000 ; CHECK IF AN 11/45 AND DETERMINE WHICH MODE AND REG, SET ARE SELECTED BY THE PSW
4890 RTT2 ; TEST IF AN 11/40
4891 024377 122737 000004 000764 CMPB #4,0#OPT,CP ; BRANCH IF NOT AN 11/40
4892 024376 001002 BNE RTT2A ; GO TO RTT2EX IF 11/40
4893 024400 000167 000200 JMP RTT2EX ; GET SAVED PSW
4894 024404 016700 177756 RTT2A1 MOV SAVPSW,R0 ; CLEAR PRIORITY LEVEL, AND COND CODES
4895 024410 105000 CLRB R0
4896 024412 012702 144000 MOV #UM+REG,R2
4897 024416 074002 XOR R0,R2
4898 024420 001435 BEQ 25 ; USER MODE REG, SET #1 ON
4899 024422 012702 044000 MOV #SM+REG,R2
4900 024426 074002 XOR R0,R2
4901 024430 001447 BEQ 35 ; SUPER MODE REG, SET #1 ON
4902 024432 032700 140000 BIT #UM,R0
4903 024436 001062 BNE RTT2EX
4904 ; TEST THAT RTT CLEARS BITS 11,12,13 & PRIORITY LEVEL BITS IN KERNEL MODE
4905 MOV #1,R2 ; KERNEL MODE REG, SET 0 ON
4906 024440 012702 177777 MOV #PUM+REG+PRTY5,0#PSW ; SELECT REG, SET #1
4907 024444 012737 034240 177776 MOV R12 ; SHOULD CLEAR REG #12
4908 024452 005002 CLR R12
4909 024454 012746 000100 MOV #PRTY2,=(SP)
4910 024460 010746 MOV PC,=(SP)
4911 024462 062716 000006 ADD #15,=(SP) ; FORM NEW PC
4912 024466 000006 RTT

```

```

DCQKCD 11/40-11/45 CPU EXERCISER MACy11 27(695) 4-SEP-74 11:53 PAGE 100
DCQKCD START OF SECTION 5
4913 024470 013700 177776 15 MOV #0#PSW,R0 ; NOW USING REG SET 0
4914 024474 005702 TST R2 ; SHOULD TEST R2 NOT R12
4915 024476 001001 BNE 45
4916 024500 104400 HLT ; ERROR!DID NOT CLEAR BIT #11 OF PSW
4917 024502 022700 000100 45 CMP #PRTY2,R0 ; TESTS THE PSW AFTER THE RTT
4918 024506 001436 BEQ RTT2EX
4919 024510 104400 HLT ; ERROR! INCORRECT PSW AFTER THE RTT
4920 024512 000434 BR RTT2EX
4921 ; TEST TO INSURE THAT RTI DOES NOT CLEAR BITS 11-15 IN USER MODE
4922 024514 052737 030340 177776 25 BIC #PUM+PRTY7,0#PSW ; PSW<15>=144X
4923 024522 005046 CLR =(SP)
4924 024524 010746 MOV PC,(SP)
4925 024526 062716 000006 ADD #55,=(SP)
4926 024532 000002 RTI ; ATTEMPS TO INSERT A PSW OF 0
4927 024534 022737 174340 177776 55 CMP #UM+PUM+REG+PRTY7,0#PSW ; SHOULD CHECK AGAINST REG #0
4928 024542 001420 BEQ RTT2EX ; ERROR! RTI CLEARED BITS IN PSW
4929 024544 104400 HLT
4930 024546 000416 BR RTT2EX
4931 ; TEST THAT BITS 11-15 AND PRIORITY BITS ARE NOT ALTERED IN SUPER MODE
4932 024550 052737 030200 177776 35 BIC #PUM+PRTY4,0#PSW ; PSW<15>=044X
4933 024556 012746 000340 MOV #PRTY7,=(SP)
4934 024562 010746 MOV PC,=(SP)
4935 024564 062716 000006 ADD #63,=(SP) ; ATTEMPS TO CLEAR 11-15 AND ALTER PRTY
4936 024570 000006 RTT
4937 024572 022737 074200 177776 65 CMP #SM+PUM+REG+PRTY4,0#PSW
4938 024600 001401 BEQ RTT2EX ; ERROR! RTT ALTERED PRTY IN
4939 024602 104400 HLT ; SUPER MODE OR BITS 11-15,
4940 RTT2EX1 MOV SAVPSW,0#PSW
4941 SCOPE
4942 MOV PC,R2
4943 ADD #12,R2 ; GO RELOCATE PROGRAM CODE
4944 MOV #RELOC,PC
4945 024604 016737 177556 177776 RTT2EX11 MOV SAVPSW,0#PSW
4946 024612 104000 SCOPE
4947 024614 010702 MOV PC,R2
4948 024616 062702 000012 ADD #12,R2
4949 024622 012707 001152 MOV #RELOC,PC ; GO RELOCATE PROGRAM CODE
4950 024626 000000 REL551,WORD 0 ; LAST ADDRESS OF CODE TO BE RELOCATED 5555555555
4951 ;SBTTL START OF SECTION 6
4952 1666666666666666 FIRST ADDRESS TO BE RELOCATED 6666666666
4953 REL61 MOV PC,R0 ; GET PC
4954 TST =(R0) ; R0 CONTAINS THE ADDRESS OF REL 6
4955 MOV #0,#FRSTAD ; SAVE
4956 #0,#FSECT ; SET SECTION #
4957 024630 010700 JSR PC,0#DDISP ; LOAD DISPLAY GEG
4958 024632 000737 001010 MOV #0#DISPL,REL66 ; GET CURRENT PC
4959 024634 010737 000000 MOV #1,R0 ; SUBTRACT RELOCATION FACTOR
4960 024636 012737 000000 SUB R0,0#FACTOR ; SAVE RELOCATION FACTOR
4961 024644 004737 005166 JSR PC,0#DISP ; SET NEW SCOPE PTR
4962 024652 013767 005172 001710 MOV PC,R1
4963 024660 010700 MOV #1,R0
4964 024662 102700 024662 SUB R0,0#FACTOR
4965 024666 010737 001024 MOV PC,R1
4966 024672 010701 MOV PC,R1

```

```

DCQKCD START OF SECTION 6
4967
4968 024674 032737 040000 000764 BIT #EISOPT,**OPT,CP ;CHECK IF 11/40 WITH EIS OPTION
4969 024702 001002 BNE ASHL0 ;BRANCH IF EIS OPT AVAIL,
4970 024704 000167 JMP MPl
4971
4972 ;CHECK ASH,ASHC,MUL, AND JIV INSTRUCTIONS
4973 024710 012700 000001 ASHL01 MOV #1,R0 ;R0 WILL BE THE SHIFT COUNT
4974 024714 012703 000021 MOV #17,R3 ;MAX SHIFT COUNT
4975 024720 005067 000014 151 CLR 25 ;PRESET SAVED CC'S LOCATION#0
4976 024724 010002 MOV R0,R2 ;GET SHIFT COUNT FOR PASS
4977 024726 010705 MOV PC,R5 ;R5 & R4 WILL BE DATA SHIFTED BY
4978 024730 010504 MOV R5,R4 ;ASH & ASL INSTRUCTIONS
4979 024732 072502 ASH R2,R5 ;SHIFT R5
4980 024734 113727 177776 MOV0 #PSW,(PC) ;SAVE CC'S
4981 024740 000000 251 .WORD 0 ;CONTAINS ASH CC'S IN EVEN BYTE
4982 ;ASL CC'S IN ODD BYTE
4983 024742 006304 351 ASL R4 ;SHIFT R4
4984 024744 113746 MOV0 #PSW,-(SP) ;SAVE PSW ON STACK
4985 024750 132716 000002 BIT0 #V,(SP) ;CHECK IF ASL SET V BIT
4986 024754 001403 BEQ 305 ;IF ASL SET V THEN SET V IN 25+1
4987 024756 152767 000002 177755 BIC0 #V,25+1 ;RESTORE ORIGINAL PSW
4988 024764 112637 177776 3051 MOV0 (SP),#PSW ;SHIFT R4 R2 TIMES
4989 024770 077214 SOB R2,35 ;SAVE CC'S AFTER ASL
4990 024772 153767 177776 177741 BIC0 #PSW,25+1 ;CHECK ASH & ASL RESULTS
4991 025000 020504 CMP R5,R4 ;
4992 025002 001004 BNE 45 ;CHECK ASH & ASL CC'S
4993 025004 126767 177730 177727 CMP0 25,25+1 ;
4994 025012 001401 BEQ ,+4 ;
4995 025014 104400 451 HLT ;ERROR! INCORRECT RESULT OR CC'S
4996 025016 005200 INC R0 ;INCREMENT PASS SHIFT COUNT
4997 025020 020003 CMP R0,R3 ;
4998 025022 001336 BNE 15 ;
4999
5000 025024 012700 177777 ASHL01 MOV #-1,R0 ;R0 = RIGHT SHIFT COUNT FOR PASS
5001 025030 012703 177757 MOV #-17,R3 ;MAX SHIFT COUNT
5002 025034 010002 151 MOV R0,R2 ;GET SHIFT COUNT FOR PASS
5003 025036 010705 MOV PC,R5 ;R5 & R4 = DATA TO BE SHIFTED
5004 025040 010504 MOV R5,R4 ;BY ASH & ASR INSTRUCTIONS
5005 025042 072502 ASH R2,R5 ;SHIFT R5 R2 TIMES
5006 025044 113727 177776 MOV0 #PSW,(PC) ;SAVE CC'S IN EVEN BYTE
5007 025050 000000 251 .WORD 0 ;CONTAINS ASH CC'S IN EVEN BYTE
5008 ;ASR CC'S IN ODD BYTE
5009 025052 005402 NEG R2 ;SHIFT R4
5010 025054 006204 351 ASR R4 ;SHIFT R4 R2 TIMES
5011 025056 077202 SOB R2,35 ;SAVE CC'S AFTER ASR
5012 025060 113767 177776 177763 MOV0 #PSW,25+1 ;ASH RIGHT WILL NOT SET V ASR MAY SET V
5013 025066 142767 000002 177755 BIC0 #V,25+1 ;CHECK ASH & ASR RESULTS
5014 025074 020504 CMP R5,R4 ;
5015 025076 001004 BNE 45 ;CHECK ASH & ASR CC'S
5016 025100 126767 177744 177743 CMP0 25,25+1 ;
5017 025106 001401 BEQ ,+4 ;
5018 025110 104400 451 HLT ;
5019 025112 005300 DEC R0 ;DECREMENT PASS SHIFT COUNT
5020 025114 020003 CMP R0,R3 ;

```

```

DCQKCD START OF SECTION 6
5021 025116 001346 BNE 15 ;
5022
5023 025120 012746 000037 ASHL01 MOV #31,(SP) ;PUT MAX SHIFT COUNT ON STACK
5024 025124 012746 000001 MOV #-1,(SP) ;PUT LEFT SHIFT COUNT ON STACK
5025 025130 011600 151 MOV (SP),R0 ;GET PASS SHIFT COUNT
5026 025132 010705 MOV PC,R5 ;CURRENT PC IS DATA TO BE SHIFTED
5027 025134 010503 MOV R5,R3 ;ASHC SHIFTS R4,R5;ASL,ROL SHIFTS R2,R3
5028 025136 005004 CLR R4 ;
5029 025140 005002 CLR R2 ;
5030 025142 073400 ASHC R0,R4 ;SHIFT R4 LEFT AS SPECIFIED BY R0
5031 025144 006303 251 ASL R3 ;SHIFT R2,R3 LEFT
5032 025146 006102 ROL R2 ;AS SPECIFIED BY R0
5033 025150 077003 SOB R0,25 ;
5034 025152 020402 CMP R4,R2 ;CHECK RESULTS
5035 025154 001002 BNE 35 ;
5036 025156 020503 CMP R0,R3 ;
5037 025160 001401 BEQ ,+4 ;
5038 025162 104400 351 HLT ;
5039 025164 005216 000002 INC0 (SP) ;INCREMENT NEXT PASS SHIFT COUNT
5040 025166 021666 CMP (SP),2(SP) ;REACHED MAX COUNT (31,)
5041 025172 001356 BNE 15 ;
5042 025174 022626 CMP (SP),+(SP) ;RESTORE STACK PTR
5043
5044 025176 012746 177740 ASHCR01 MOV #32,(SP) ;PUT MAX RIGHT SHIFT COUNT ON STACK
5045 025202 012746 177777 MOV #-1,(SP) ;PUT PASS SHIFT COUNT ON STACK
5046 025206 011600 151 MOV (SP),R0 ;GET PASS SHIFT COUNT
5047 025210 010702 MOV PC,R2 ;R2,R3 & R4,R5 ARE THE DATA REGISTERS
5048 025212 010204 MOV R2,R4 ;TO BE SHIFTED BY TEST
5049 025214 005003 CLR R3 ;
5050 025216 005005 CLR R5 ;
5051 025220 000262 SEV ;SET V BIT IN PSW
5052 025222 073200 ASHC R0,R2 ;SHIFT R2,R3 RIGHT R0 TIMES
5053 025224 102410 BVS 35 ;SHIFT RIGHT CLEARS V
5054 025226 005400 NEG R0 ;NEGATE SHIFT COUNT FOR SOB
5055 025230 006204 251 ASR R4 ;SHIFT R4,R5 RIGHT R0 TIMES
5056 025232 006005 ROR R5 ;
5057 025234 077003 SOB R0,25 ;
5058 025236 020204 CMP R0,R4 ;CHECK RESULT
5059 025240 001002 BNE 35 ;
5060 025242 020305 CMP R3,R5 ;
5061 025244 001401 BEQ ,+4 ;
5062 025246 104400 351 HLT ;
5063 025250 005316 DEC (SP) ;SET SHIFT COUNT FOR NEXT PASS
5064 025252 021666 CMP (SP),2(SP) ;CHECK IF MAX SHIFT COUNT
5065 025256 001353 BNE 15 ;
5066 025260 022626 CMP (SP),+(SP) ;RESTORE STACK PTR
5067
5068 SCOPE
5069 ;THE BELOW TEST OF THE MUL INSTRUCTION MULTIPLIES THE CURRENT PC
5070 ;BY 1,2,4,8 ETC AND SHIFTS THE SAME PC VALUE USING AN ASHC LEFT BY
5071 ;10,1,2,3,ETC AND COMPARES THE RESULTS, CONDITION CODE RESULTS ARE NOT CHECKED,
5072 ;R0 CONTAINS MULTIPLIER FOR MUL
5073 ;R1,2,3,ETC
5074 ;R5,R4 ARE DATA REGISTERS

```

DC0K0C START OF SECTION 5

```

5075 025274 010227      MOV      R2,(PC)+      ;SAVE MULTIPLICAND
5076 025276 000000      ,WORD      0          ;CONTAINS ORIGINAL MULTIPLICAND
5077 025300 005003      CLR      R3
5078 025302 005004      CLR      R4
5079 025304 010205      MOV      R2,R5        ;FORM MUL AND ASHC
5080 025306 100001      BPL      ,+4          ;IF MULTIPLICAND IS NEG THEN SET R4 = -1
5081 025310 005104      COM      R4           ;FOR ASHC
5082 025312 000277      SCC      ;PRESET CC'S
5083 025314 070200      MUL      R0,R2        ;MULTIPLY R2 BY R0 LEAVE PRODUCT
5084 025316 102406      BVS      2$          ;IN R2,R3 MSH IN R2,LSH IN R3
5085 025320 001405      BEQ      2$          ;PRODUCT WILL NEVER BE = 0
5086 025322 073416      ASHC     (SP),R4      ;MULTIPLY R4,R5 BY (SP) LEAVE PRODUCT
5087 025324 000000      ;IN R4,R5 MSH IN R4,LSH IN R5
5088 025326 020204      CMP      R2,R4        ;CHECK MSH RESULT
5089 025328 001002      RNE      2$
5090 025330 020305      RNE      2$
5091 025332 001401      BEQ      R3,R5        ;CHECK LSH RESULT
5092 025334 004400      ,+4
5093 025336 005216      HLT      (SP)        ;INCREMENT ASHC SHIFT COUNT
5094 025338 006300      INO      R0          ;SHIFT MUL MULTIPLIER
5095 025340 006300      ASL      1$
5096 025342 102353      BVC      ;CHECK MUL INST WITH MULTIPLIER (R0) = 100000
5097 025344 010702      MOV      PC,R2        ;R2 = MULTIPLICAND
5098 025346 005202      INC      R2
5099 025348 010227      MOV      R2,(PC)+      ;SAVE MULTIPLICAND
5100 025350 000000      ,WORD      0          ;CONTAINS ORIGINAL MULTIPLICAND
5101 025352 000003      COM      R3
5102 025354 005103      MOV      R2,R4        ;R4 WILL BE MSH 'PRODUCT'
5103 025356 010204      ASR      R4           ;FORM 'PRODUCT'
5104 025358 006204      COM      R4           ;COMPLEMENT MSH 'PRODUCT'
5105 025360 005104      MUL      R0,R2        ;MULTIPLY R2 BY 100000 LEAVING
5106 025362 070200      ;R2 = MSH, R3 = LSH PRODUCT
5107 025364 000000      ;COMPARE MSH PRODUCTS
5108 025366 020204      CMP      R2,R4
5109 025368 001002      BNE      3$
5110 025370 020003      CMP      R0,R3        ;CHECK LSH PRODUCT
5111 025372 001401      BEQ      ,+4
5112 025374 104400      HLT
5113 025376 104400      SCOPE
5114
5115
5116 ;THE BELOW TEST OF THE DIV INSTRUCTION DIVIDES THE CURRENT PC BY
5117 ;1,2,4,8,ETC LEAVING THE QUOTIENT/REMAINDER IN R2/R3, NEXT THE QUOTIENT
5118 ;IS MULTIPLIED BY 1,2,4,8,ETC AND THE REMAINDER ADDED, THE RESULT IS
5119 ;THEN COMPARED WITH THE ORIGINAL CURRENT PC,
5120 DIV0: MOV      #1,R0      ;R0=DIVISOR
5121 025400 010737 000001      MOV      PC,#100      ;SAVE DATA IN 10$
5122 025402 013703 025500      MOV      #100,R3      ;GET DATA
5123 025404 005002      CLR      R2           ;CLEAR MSH DIVIDEND
5124 025406 000277      SCC
5125 025408 071200      DIV      R0,R2        ;DIVIDE R2 BY R0 LEAVING QUOTIENT IN R2
5126 025410 103421      ;AND REMAINDER IN R3
5127 025412 100420      BCS      2$
5128 025414 102010      BHI      2$
5129 025416 102010      BVC      20$         ;BRANCH IF DIVIDE WORKED

```

DC0K0C START OF SECTION 6

```

5129 025432 022700 000001      CMP      #1,R0        ;V BIT SHOULD ONLY SET IF DIVIDING BY 1
5130 025434 001014 025500      BNE      2$          ;AND THE LSH OF DIVIDEND
5131 025436 032737 100000 025500      BIT      #10000000010$ ;IS NEGATIVE
5132 025438 001410      BEQ      3$
5133 025440 000410      BR      3$
5134 025442 010204      MOV      R2,R4        ;GET QUOTIENT
5135 025444 070400      MUL      R0,R4        ;MULTIPLY QUOTIENT BY DIVISOR
5136 025446 060305      ADD      R3,R5        ;ADD REMAINDER TO LSH PRODUCT
5137 025448 103403      BCS      2$          ;SHOULD BE NO CARRY
5138 025450 023705 025500      CMP      #0100,R5     ;CHECK RESULT
5139 025452 001401      BEQ      ,+4
5140 025454 104400      HLT
5141
5142
5143
5144 025472 006300      3$: ASL      R0
5145 025474 102346      BVC      1$
5146 025476 000401      BR      ASHL1
5147 025500 000000      10$: ,WORD      0          ;CONTAINS ORIGINAL PC USED AS DATA
5148
5149
5150 ;CHECK ASH,ASHC,MUL, AND DIV INSTRUCTIONS USING ADDRESS MODE 1
5151 025502 005016      ASHL1: (SP)
5152 025504 005000      CLR      R0          ;(SP) = SHIFT COUNT
5153 025506 012702 000020      MOV      #16,R2      ;R0 = SHIFT COUNT FOR CHECK ASH
5154 025508 000012      CLR      R2          ;R2 = MAX LEFT SHIFT COUNT
5155 025510 010703      MOV      PC,R3        ;CLEAR CC'S HOLDING ADDRESS
5156 025512 000003      MOV      R3,R4        ;R3,R4 = DATA TO BE SHIFTEC
5157 025514 072316      MOV      (SP),R3     ;SHIFT R3 LEFT (SP) TIMES
5158 025516 013727 177776      MOV      #0,PSH,(PC)+ ;SAVE CC'S
5159 025518 000000      ,WORD      0          ;CONTAINS ASH (SP),R3 CC'S IN EVEN BYTE
5160 025520 072400      ASH      R0,R4        ;AND ASH R0,R4 CC'S IN ODD BYTE
5161 025522 113767 177776 177767      MOV      #0,PSH,2$+1 ;SHIFT R4 LEFT R0 TIMES
5162 025524 020304      CMP      R3,R4        ;SAVE CC'S IN ODD BYTE OF 2$
5163 025526 001004      BNE      3$          ;COMPARE RESULTS
5164 025528 126767 177756 177755      CMPB    2$,2$+1      ;BRANCH IF THEY DO NOT COMPARE
5165 025530 001401      BEQ      ,+4          ;CHECK CC'S AFTER ASH INSTRUCTIONS
5166 025532 104400      HLT
5167
5168 ;ERROR! EITHER RESULTS OF SHIFT OR
5169 ;RESULT CC'S ARE INCORRECT
5170 025534 005200      INC      R0          ;INCREMENT SHIFT COUNT FOR ASH R0,R4
5171 025536 005216      INC      R0          ;INCREMENT SHIFT COUNT FOR ASH (SP),R3
5172 025538 020200      CMP      R2,R0        ;CHECK FOR MAX SHIFT COUNT
5173 025540 001351      BNE      1$
5174 025542 005016      ASHR1: CLR      (SP)
5175 025544 005000      CLR      R0          ;(SP) = SHIFT COUNT FOR ASH (SP),R4
5176 025546 005402      CLR      R2          ;R0 = SHIFT COUNT FOR ASH R0,R5
5177 025548 005000      NEG      R2          ;R2 = MAX RIGHT SHIFT COUNT (SET BY
5178 025550 005000      ;ABOVE TEST TO 16, NOW = -16,
5179 025552 005000      ;CLEAR CC'S HOLDING ADDRESS
5180 025554 005000      ;R4,R5 = DATA TO BE SHIFTEC RIGHT
5181 025556 005000      MOV      R4,R5
5182 025558 072416      MOV      (SP),R4     ;SHIFT R4 RIGHT (SP) TIMES
5183 025560 013727 177776      MOV      #0,PSH,(PC)+ ;SAVE CC'S
5184 025562 000000      ,WORD      0          ;CONTAINS ASH (SP),R4 CC'S IN EVEN BYTE

```



```

DC0KCD 11/40-11/45 CPU EXERCISER MACY11 27(655) 4-SEP-74 11:53 PAGE 107
DC0KCD START OF SECTION 6
5291 026246 032737 14 020 177776 MPI; BIT #UM,#PSW ;KERNEL MODE?
5292 026254 001537 BEQ ENDCP ;YES EXIT TEST
5293 026256 010746 MOV PC,(SP)
5294 026260 062716 000144 ADD #5,(SP)
5295 026264 012637 000250 MOV (SP),#HMVEC ;SET MEM MGMT ABORT VECTOR
5296 026270 005046 CLR =(SP) ;CLEAR CMECK WORD
5297 026272 010603 MOV SP,R3
5298 026274 010346 MOV R3,(SP) ;PUT ADDRESS OF CHECK WORD ON THE STACK
5299 026276 105737 000770 YSTB #MMON ;CHECK IF MEM MGMT IS ENABLED
5300 026302 001423 BEQ 1$ ;BRANCH IF OFF
5301 026324 013737 177640 177654 MOV #UISPAR0,#UIPAR0 ;SET UP USER PAGE ADDR, REG,
5302 026312 012737 006006 177614 MOV #0006,#UIPDR ;SET USER PAGE DESC REG R/W UP 6 PAGES
5303 026320 122737 000004 000764 CMPB #4,#OPTY,CP ;BRANCH IF 11/40
5304 026326 001406 BEQ 10$
5305 026330 013737 172240 172254 MOV #SIPAR0,#SIPAR0
5306 026336 012737 006006 172214 MOV #0006,#SIPDR ;SET SUPER PAGE DESC, REG,
5307 026344 062706 140000 #140000,SP ;SET CURRENT MODE'S STACK POINTER
5308 026350 000240 NOP
5309 026352 010746 1$; MOV PC,(SP)
5310 026354 062716 000024 ADD #3,(SP)
5311 026360 012637 000030 MOV (SP+),#EMTVEC ;SET EMT TRAP VECTOR
5312 026364 104000 EMT ;TRAP TO 3$ BELOW
5313 026366 005266 000002 INC 2(SP) ;INCREMENT CHECK WORD
5314 026372 001417 BEQ 6$
5315 026374 104400 4$; HLT ;ERROR! MFPI,MPI FAILURE-FOR BETTER
5316 026376 000415 BR 6$ ;ISOLATION SUGGEST RUNNING MFPI DIAG, DCKTD/E
5317 026400 000240 3$; NOP ;PSW=KERNEL MODE,PREV USER OR SUPER MODE
5318 026402 006506 MFPI SP ;GET PREV. MODE'S STACK POINTER
5319 026404 006536 MFPI 0(SP)+ ;GET DATA (AN ADDRESS) ON PREV MODE'S STACK
5320 026406 006576 MFPI 0(80) ;GET DATA (00) FROM PREV MODE'S ADDRESS
5321 026412 000240 NOP ;SPACE AND PUSH ONTO KERNEL STACK
5322 026414 001367 BNE 4$ ;ERROR IF BRANCH TAKEN! SHCULD HAVE A ZERO ON THE STACK
5323 026416 005116 COM 1(SP) ;COMPLEMENT OPERAND
5324 026420 006636 MTP 0(80)+ ;POP OPERAND OFF KERNEL STACK AND MOVE
5325 ;IT TO PREV MODE'S SPACE
5326 026422 000002 RTI ;RETURN TO INST FOLLOWING EMT ABOVE
5327 026424 104400 5$; HLT ;ERROR! MEMORY HANG, ABORT
5328 026426 105037 177776 CLR #PSW ;SET PRIORITY LEVEL BACK TO 0
5329 026432 012737 005244 000250 6$; MOV #KTABRT,#HMVEC ;RESTORE VECTOR
5330 026440 012737 001014 000030 MOV #SCGPEA,#EMTVEC
5331 026446 012706 000500 MOV #STKPTR,SP ;RESTORE STACK POINTER
5332 026452 104000 SCOPE
5333 ;CHECK THAT WALT INSTRUCTION TRAPS TO 4 (11/45),10 (11/40) IN USER/SUPER MODE
5334 HALT; MOV PC,(SP) ;GET CURRENT PC
5335 026454 010746 ADD #2,(SP)
5336 026456 062716 000022 MOV (SP),#ERRVEC ;SET ERROR TRAP VECTOR TO 2$ BELOW
5337 026462 011637 000004 MOV (SP+),#RESVEC ;LOAD RESERVED INST TRAP VECTOR (11/40)
5338 026466 012637 000010 MOV (SP+),#RESVEC ;SHOULD TRAP TO 4 IN USER/SUPER MODE
5339 026472 000000 HALT ;ERROR! WALT ABOVE FAILED IN USER/SUPER MODE
5340 026474 104400 1$; HLT
5341 026476 000404 BR 3$
5342 026500 010716 2$; MOV PC,(SP) ;REPLACE RETURN PC WITH
5343 026502 062716 000006 ADD #3,(SP) ;ADDRESS OF 3$ BELOW
5344 026506 000002 RTI ;RETURN (TO 3$)

```

```

DC0KCD 11/40-11/45 CPU EXERCISER MACY11 27(655) 4-SEP-74 11:53 PAGE 108
DC0KCD START OF SECTION 6
5345
5346 026510 012737 005274 000004 3$; MOV #ERRRT,#ERRVEC ;RESTORE ERROR TRAP VECTOR
5347 026516 012737 005264 000010 MOV #RESERR,#RESVEC
5348 026524 104000 SCOPE
5349
5350 ;TEST THAT RESET IS A 'NOP' IN USER/SUPER MODE
5351 026526 000277 RESET; SCC
5352 026530 013700 177776 MOV #PSW,R0 ;GET CURRENT PSW
5353 026534 000277 SCC #PSW,R0
5354 026536 000005 RESET
5355 026540 023700 177776 CMP #PSW,R0 ;CHECK THAT PSW UNCHANGED BY RESET ABOVE
5356 026544 001401 BEQ ++
5357 026546 104400 HLT ;ERROR! RESET CLEARED MODE BITS IN PSW
5358 026550 010017 177776 MOV R0,#PSW ;RESTORE PSW (FOR ERROR)
5359 026554 104000 ENDCP; SCOPE
5360
5361 026556 010702 MOV PC,R2
5362 026560 062702 000012 ADD #12,R2
5363 026564 012707 001152 MOV #RELOC,PC ;GO RELOCATE PROGRAM CODE
5364 026570 001000 REL66; ;WORD 0
5365 ;6666666666666666 LAST ADDRESS OF CODE TO BE RELOCATED 666666666666
5366
5367 ;SBTTL START OF SECTION 7
5368 ;7777777777777777 FIRST ADDRESS TO BE RELOCATED 7777777777
5369 026572 010700 REL7; MOV PC,R0 ;GET PC
5370 026574 005740 TST =(R0) ;R0 CONTAINS THE ADDRESS OF REL7
5371 026576 010037 001010 MOV #R,#RSTAD ;SAVE
5372 026602 012737 000007 #7,#SECT ;SET SECTION #
5373 026610 004737 009166 JSR PC,#DDISP ;LOAD DISPLAY BEG
5374 026614 013767 009172 #DDISPLY,REL7
5375 026622 010700 MOV PC,R0 ;GET CURRENT PC
5376 026624 102700 026624 #,R0 ;SUBTRACT RELOCATION FACTOR
5377 026630 010037 001004 MOV R0,#FACTOR ;SAVE RELOCATION FACTOR
5378 026634 010701 MOV PC,R1 ;SET NEW SCOPE PTR
5379
5380 ;TEST STACK LIMIT REGISTER
5381 ;THIS TEST SHIFTS A '1' BIT THROUGH ALL BIT POSITIONS
5382 026636 032737 004000 000764 STKLIM; BIT #KJOPT,#OPTY,CP ;CHECK IF OPTION IS AVAILABLE
5383 026644 001512 BEQ 1015 ;EXIT IF NOT AVAILABLE
5384 026646 012702 177774 MOV #SLR,R2 ;GET ADDRESS OF STACK LIM REG
5385 026652 005022 CLR (R2)+ ;CLEAR STACK LIMIT REG
5386 026654 032712 000020 BIT #1,(R2) ;EXIT TEST IF '1' BIT IS SET
5387 026660 001104 BNE 1015
5388 026662 052712 000340 BIS #340,(R2) ;SET PRIORITY LEVEL 7 TO PREVENT
5389 ;ANY INTERRUPTS FROM OCCURRING
5390 026666 012700 000400 MOV #400,R0 ;SET CHECK DATA
5391 026672 010042 1$; MOV R0,(R2) ;MOVE TO STACK LIMIT REG
5392 026674 022200 CMP #1,(R0) ;AND CHECK RESULT
5393 026676 001401 BEQ 2$
5394 026700 104400 HLT ;ERROR! STACK LIMIT DID NOT
5395 ;LOAD CORRECTLY, CORRECT RESULT
5396
5397 026702 006300 2$; ASL R0 ;IS IN R0
5398 026704 103372 RCC 1$ ;SHIFT '1' BIT LEFT
5399 ;LOOP UNTIL 1 BIT SHIFTS OUT

```

```

5399 026726 25042 CLR (R2) ICLEAR STACK LIMIT REG
5400
5401 ;THIS TEST CHECKS THAT A PROPER 'RED' ZONE VIOLATION OCCURS; NOTE THAT
5402 ;NO 'RED ZONE' VIOLATION WILL OCCUR IF IN USER/SUPER MODES;
5403 ;A RED ZONE VIOLATION PUSHES THE CURRENT PSW,PC ON A STACK AT 2 AND 0
5404 ;AND TAKES THE NEXT INSTRUCTION FROM THE PC IN LOCATION 4. THE INST-
5405 ;RUCTION CAUSING THE RED ZONE VIOLATION IS 'ABORTED'.
5406 026710 010746 MOV PC,(SP) IGET CURRENT PC
5407 026712 062716 000054 ADD #45,(SP) IFORM ADDRESS OF 43 BELOW
5408 026716 012637 000004 MOV (SP+,#ERRVEC ISET ERROR TRAP VECTOR TO 45 BELOW
5409 026722 013737 177776 000006 MOV #PSW,#ERRVEC+2 IRETAIN CURRENT STATUS ON TRAP
5410 026730 010712 MOV PC,(R2) ISET STACK LIMIT TO CURRENT PC
5411 I+400
5412 026732 011206 MOV (R2),SP IAND STACK PTR = STACK LIMIT REG
5413 026734 010603 MOV SP,R3 ISAVE STACK PTR
5414 026736 0163P4 000336 MOV 336(R3),R4 ISAVE MEMORY LOC CONTENTS
5415 IAT 'RED ZONE' BOUNDARY
5416 026742 032771 140000 177776 BIT #UM,#PSW IBRANCH IF IN KERNEL MODE
5417 026750 001403 BEQ 205
5418 026752 010466 000336 MOV R4,#36(SP) ISHOULD NOT CAUSE TRAP
5419 026756 000470 BR 1005
5420
5421 026760 005066 000336 205I CLR 336(SP) ISHOULD CAUSE 'RED ZONE' TRAP
5422 026764 104400 35I HLT IERROR! FAILED TO TRAP
5423
5424 026766 032737 140000 000002 45I BIT #UM,#2 ICHECK IF TRAPPED WHEN IN USER
5425 ;/SUPER MODES (2 CONTAINS OLD PSW)
5426 026774 001013 BNE 995 IGO TO ERROR CALL
5427 026776 010600 MOV SP,R0 ISTACK PTR SHOULD = 0
5428 027000 001011 RNE 995 IGO TO ERROR CALL IF NOT 0
5429 027002 026304 000336 CMP 336(R3),R4 ICHECK THAT INST WAS ABORTED
5430 027006 001006 BNE 995 IGO REPORT ERROR
5431 027010 005012 55I CLR (R2) ICLEAR STACK LIMIT REG
5432 027012 010705 MOV PC,R5 IGET CURRENT PC
5433 027014 062705 177750 ADD #35,(R5) IFORM ADDRESS OF 35 ABOVE
5434 027020 020516 CMP R5,(SP) ICHECK THAT RETURN PC IS ON
5435 ;THE STACK (AT 0)
5436 027022 001406 BEQ 1005 IEXIT TEST
5437
5438 IERROR
5439 027024 005012 995I CLR (R2) ICLEAR STACK LIMIT REG
5440 027026 010463 000336 MOV R4,#36(R3) IRESTORE MEM LOCATION
5441 027032 012706 000500 MOV #STKPTR,SP ISET STACK PTR
5442 027036 104400 HLT IERROR!
5443 027040 010463 000336 1005I MOV R4,#36(R3) IRESTORE MEM LOCATION
5444 027044 005022 CLR (R2) ICLEAR STACK LIM REG
5445 027046 012706 000500 MOV #STKPTR,SP ISET STACK PTR
5446 027052 042712 000340 BIC #340,(R2) ISET PRIORITY LEVEL BACK TO 0
5447 027056 012737 005274 000004 MOV #ERRPTR,#ERRVEC IRESTORE ERROR TRAP VECTOR
5448 027064 012737 000002 000006 MOV #RTI,#ERRVEC+2
5449 027072 104000 1015I SCOPE
5450
5451 IMEMORY MANAGEMENT REGISTER TESTS
5452 IPDR TEST - THIS TEST WRITES/READS A COMPLEMENTING BINARY
    
```

```

5453 ICOUNT PATTERN INTO ALL PDR REGISTERS
5454
5455 027074 005737 000764 KTPDR: TST #0RT,Cp IEXIT TEST IF NO KT OPTION
5456 027100 100124 RPL KT1
5457 027102 105737 000770 TSTB #MMON IEXIT TEST IF KT IS ENABLED
5458 027106 001121 BNE KT1
5459 027110 012702 027316 MOV #PDRtbl,R2 ISET TABLE ADDRESS OF PDR'S
5460 027114 012705 100360 MOV #100360,R5 ISET BIT MASK (11/45)
5461 027120 122737 000004 000764 CMPB #4,#OPT,Cp IBRANCH IF 11/45
5462 027126 001005 IS
5463 027130 005062 000004 CLR 4(R2) ITERMNATE TABLE AT SIPDR0
5464 027134 005062 000022 CLR 22(R2)
5465 027140 005205 15I INC R5 ISET BIT MASK (11/40)
5466 027142 012200 MOV (R2)+,R0 IGET PDR ADDRESS
5467 027144 001423 BEQ 1005 IEXIT ON '0' TERMINATOR
5468 027146 012703 000010 25I MOV #0,R3 ISET LOOP COUNT (FOR 8 REGS)
5469 027152 005004 35I CLR R4 IINITIALIZE DATA TO BE WRITTEN
5470 027154 040504 45I BIC R5,R4 ICLEAR NON-SETTABLE BITS
5471 027156 010410 MOV R4,(R0) IWRITE INTO PDR
5472 027160 021004 CMP (R0),R4 IAND CHECK DATA READ BACK
5473 027162 001012 BNE 995 IGO TO ERROR CALL
5474 027164 005104 COM R4 ICOMPLEMENT DATA
5475 027166 040504 BIC R5,R4 ICLEAR NON-SETTABLE BITS
5476 027170 010410 MOV (R0),R4 IWRITE COMPLEMENT DATA INTO PDR
5477 027172 021004 CMP (R0),R4 IAND CHECK
5478 027174 001005 BNE 995 IGO TO ERROR CALL
5479 027176 005404 NEG R4 ISTEP DATA
5480 027200 100365 BPL 45 IAND LOOP UNTIL BINARY COUNT
5481 IFINISHED
5482 027202 005020 55I CLR (R0)+ ISTEP TO NEXT REGISTER
5483 027204 077316 SOB R3,#5 IUNTIL 8 REGISTERS ARE DONE
5484 027206 000755 BR 15 IGET NEXT SET OF 8 REGISTERS
5485 027210 124400 995I HLT IERROR! INCORRECT DATA READ
5486 IBACK FROM PDR, ADDRESS OF
5487 IPDR IS IN R0, DATA IS IN R4
5488 027212 000773 BR 55 ISTEP TO NEXT REGISTER
5489 027214 104020 1005I SCOPE
5490
5491 IPAR TEST - THIS TEST WRITES/READS A COMPLEMENTING BINARY COUNT
5492 IPATTERN INTO ALL PAR REGISTERS,
5493 027216 012702 027334 KTPAR: MOV #PARTBL,R2 IGET TABLE ADDRESS OF PAR'S
5494 027222 012705 170000 MOV #170000,R5 ISET BIT MASK
5495 027226 122737 000010 000764 CMPB #10,#OPT,Cp
5496 027234 001001 BNE IS
5497 027236 005005 CLR R5
5498 027240 012200 MOV (R2)+,R0 IGET PAR ADDRESS
5499 027242 001423 BEQ 1005 IEXIT ON '0' TERMINATOR
5500 027244 012703 000010 25I MOV #0,R3 ISET LOOP COUNT (FOR 8 REGS,)
5501 027250 005004 35I CLR R4 IINITIALIZE DATA
5502 027252 040504 45I BIC R5,R4 ICLEAR NON-SETTABLE BITS
5503 027254 010410 MOV R4,(R0) IWRITE INTO PAR
5504 027256 021004 CMP (R0),R4 IAND CHECK
5505 027260 001012 BNE 995 ITAKE ERROR EXIT
5506 027262 005104 COM R4 ICOMPLEMENT DATA
    
```

```

5507 027264 040504 BIC R5,R4 ;CLEAR NON-SETTABLE BITS
5508 027266 010410 MOV R4,(R0) ;WRITE COMPLEMENT DATA
5509 027270 021004 CMP (R0),R4 ;AND CHECK
5510 027272 001005 RNE 99S ;TAKC ERROR EXIT
5511 027274 005404 NEG R4 ;STEP DATA
5512 027276 100365 BPL 4S ;LOOP UNTIL FINISHED
5513
5514 027300 005020 551 CLR (R0)+
5515 027302 077310 SOB R3,0S
5516 027304 000755 BR 1S
5517
5518 027306 104400 99S: HLT ;ERROR! INCORRECT DATA READ BACK
5519 ;FROM PAR, ADDRESS OF PAR IS IN
5520 ;R0, DATA IS IN R4
5521 027310 000773 BR 5S ;DO NEXT REGISTER
5522 027312 104000
5523 027314 000416 100S: SCOPE
5524 BR KT1
;TABLES FOR PDR & PAR TESTS ABOVE
;PARTBL1
;WORD KIPDR0
;WORD UIPDR0
;WORD SIPDR0 ;CHANGED TO '0' IF 11/40
;WORD KDPDR0
;WORD UDPDR0
;WORD SDPDR0
;WORD 0 ;TERMINATOR
;PARTBL1
;WORD KIPAR0
;WORD UIPAR0
;WORD SIPAR0 ;CHANGED TO '0' IF 11/40
;WORD KDPAR0
;WORD UDPAR0
;WORD SDPAR0
;WORD 0 ;TERMINATOR
5540
5541 027352 105737 000770 KT1: TSTB @#MMON ;BRANCH IF MEM MGMT NOT
5542 027356 021522 BEQ KTEX ;ENABLED
5543 027360 005037 172350 CLR @#KIPAR4 ;SET UP MEM MGMT REGISTERS
5544 027364 005037 172310 CLR @#KIPDR4 ;TO ABORT IF A MEMORY
5545 027370 005037 177650 CLR @#UIPAR4 ;REFERENCE IS MADE TO
5546 027374 005037 177610 CLR @#UIPDR4 ;ADDRESSES (VIRTUAL) BETWEEN
5547 027400 122737 000004 000764 CMPB #4,@#OPT,CP ;100000-117776 IN ALL MODES
5548 027406 001404 BEQ 1S
5549 027410 005037 172250 CLR @#SIPAR4
5550 027414 005037 172210 CLR @#SIPDR4
5551 027420 013746 000250 1S: MOV @#MMVEC,@(SP) ;SAVE MEM MGMT VECTOR
5552 027424 013746 000252 MOV @#MMVEC+2,@(SP) ;AND PRIORITY
5553 027430 010746 MOV PC,@(SP) ;SET MEM MGMT
5554 027432 002716 000040 ADD #4,@(SP) ;VECTOR TO 4S BELOW
5555 027436 012637 000250 MOV @#PSW,@#MMVEC
5556 027442 013737 177776 000252 MOV @#PSW,@#MMVEC+2
5557 027450 005000 CLR R0 ;CLEAR ABORT INDICATOR
5558 027452 010702 MOV PC,R2 ;SET R2 AND R3 NOTE!
5559 027454 012703 100000 MOV @#0000,R3 ;THE REF VIA R3 CAUSES THE
5560 027460 014223 2S: MOV @(R2),(R3)+ ;ABORT
    
```

```

5561 027462 005700 3S: TST R0 ;BRANCH IF THE ABORT OCCURRED
5562 027464 001001 BNE ,+4
5563 027466 104400 HLT ;REPORT ERROR
5564 027470 000451 BR 100S
5565 ;ABORT HERE
5566 027472 013700 177776 4S: MOV @#PSW,R0 ;SR0 SHOULD CONTAIN
5567 027476 000300 SWAB R0 ;CAUSE FOR ABORT AND
5568 027500 006200 ASR R0 ;ALSO WHICH SEGMENT
5569 027502 042700 BIC #179637,R0 ;WAS IN USE WHEN ABORT
5570 027506 062700 100011 ADD #100011,R0 ;OCCURRED,
5571 027512 020037 177572 CMP R0,@#SR0
5572 027516 001031 BNE 99S
5573 027520 012700 027460 MOV @2S,R0 ;GET ADDRESS OF INST THAT ABORTED
5574 027524 020037 177576 CMP R0,@#SR2 ;THAT ABORTED
5575 027530 001024 BNE 99S
5576 027532 122737 000004 000764 CMPB #4,@#OPT,CP
5577 027540 001414 BEQ 5S
5578 027542 012700 000362 MOV #362,R0
5579 027546 120037 177574 CMPB R0,@#SR1 ;SR1 (11/45) CONTAINS REGISTER
5580 027552 001013 BNE 99S ;MODIFICATIONS MADE
5581 027554 012700 000023 MOV @2S,R0
5582 027560 120037 177575 CMPB R0,@#SR1+1
5583 027564 001006 BNE 99S
5584 027566 012700 027460 MOV @2S,R0
5585 027572 005720 551: TST (R0)+ ;R0=ADDRESS OF INST FOLLOWING ABORT
5586 027574 020016 CMP R0,(SP) ;(3S)
5587 027576 001001 BNE 99S ;RETURN
5588 027600 000002 RTI
5589 ;ENTER HERE ON ERROR
5590 027602 104400 99S: HLT ;REPORT ERROR
5591 027604 010716 MOV PC,(SP)
5592 027606 062716 177654 ADD #3S,@(SP)
5593 027612 000002 RTI ;RETURN
5594 027614 012637 000252 100S: MOV (SP)+,@#MMVEC+2 ;RESTORE ABORT VECTOR
5595 027620 012637 000250 MOV (SP)+,@#MMVEC ;& PRIORITY,
5596 027624
5597 KTEX:
5598 027624 010702 MOV PC,R2
5599 027626 062702 000012 ADD #12,R2
5600 027632 012707 001152 MOV @RELOC,PC ;GO RELOCATE PROGRAM CODE
5601 027636 000000 REL77: WORD 0
5602 ;7777777777777777 LAST ADDRESS OF CODE TO BE RELOCATED 77777777777
5603
5604 ;SOFTL TELETYPE & CLOCK TESTS
5605 ;CHECK TTY INTERRUPT
5606 027640 005037 001004 TTYCHK: CLR @#FACTOR
5607 027644 010701 MOV PC,R1
5608 027646 032737 000400 000764 BIT *TTYP:,@#OPT,CP ;BRANCH IF CONSOLE TTY AVAIL
5609 027654 001002 BNE 1S
5610 027656 000167 000556 JMP ARRFIN ;JUMP IF NOT AVAILABLE
5611 027662 032737 000100 151: BIT #100,@#TPS ;CHECK IF TTY IS READY
5612 027670 001374 RNE ,+6
5613 027672 012737 027746 000064 MOV #3S,@#TPVEC ;SET TTY INTERRUPT VECTOR
5614 027700 012737 022200 000066 MOV #2R0,@#TPVEC+2 ;PRIORITY LEVEL 4 ON INTERRUPT
    
```

```

DC0KCD TELETYPE CLOCK TESTS
5615 027726 0 7767 030034 000114 MOV #NULLS,MSG ;ADDRESS OF MESSAGE TO BE TYPED
5616 027714 1 7737 000110 177566 MOVB #MSG,#TPB ;TYPE FIRST CHARACTER OF MESSAGE
5617 027722 1 5737 177564 TSTB @#TPS
5618 027726 1 0375 BPL ;
5619 027730 0 6237 177564 ASR #TPS ;SET IE BIT IN TTY CSR REG
5620 027734 0 0001 WAIT ;WAIT FOR FIRST INTERRUPT
5621 027736 0 0440 BR KW11
5622 027740 0 6337 177564 ASL @#TPS ;CLEAR IE BIT
5623 027744 0 0002 RTI
5624
5625 027746 122777 000012 000054 351 CMPB #12,#MSG ;CHECK IF CHARACTER IS <L>
5626 027754 0 0102 BNE 45
5627 027756 0 6337 177564 ASL @#TPS ;CLEAR IE BIT
5628 027762 0 5237 000340 177776 BIS #PR17,@#PSH ;SET PRIORITY LEVEL 7
5629 027770 0 13746 177776 MOV #PSW,(SP) ;PUSH PSW ONTO STACK
5630 027774 0 04767 152564 JSR PC,TYPE
5631 030000 0 00752 CRLF
5632 030002 0 5237 000100 177564 BIS #100,@#TPS ;SET IE BIT
5633 030010 0 05267 000014 MSG ;STEP POINTER
5634 030014 0 00002 RTI
5635 030016 1 17737 000006 17-566 451 MOVB @MSG,@#TPB ;TYPE CHARACTER
5636 030024 0 01745 BEQ 25 ;BRANCH IF TERMINATOR
5637 030026 0 05227 551 INC (PC)* ;SET MSG TO NEXT CHAR ADDRESS
5638 030030 0 00000 MSG ;CONTAINS ADDRESS OF CHAR TO BE TYPED
5639 030032 0 00002 RTI ;RETURN
5640 030034 0 20015 000015 NULLS ;ASCIZ <15><40><15> ;CAR RET,SPACE,CAR RET.
5641
5642
5643 ;ROUTINE TO TURN ON KW11-P OR KW11=L LINE CLOCK IF AVAILABLE
5644 030040 0 10701 KW11 MOV PC,R1
5645 030042 0 12737 030350 000100 MOV #LKSRV,#LKVEC ;LOAD INTERRUPT VECTOR
5646 030050 0 12737 030400 000104 MOV #PLKSRV,#PLKVEC ;FOR KW11-L & KW11-P CLOCKS
5647 030056 0 12737 000300 000102 MOV #300,@LKVEC+2 ;SET PRIORITY LEVEL 6 ON INT.
5648 030064 0 12737 000300 000106 MOV #300,@PLKVEC+2
5649 030072 0 32737 002000 001754 BIT #LKOPT,@#OPT,CP ;BRANCH IF 'P' CLOCK NOT AVAIL
5650 030100 0 01407 BEQ 105
5651 030102 0 12737 000002 172542 MOV #2,@#PLKCSB ;LOAD COUNT SET BUFFER
5652 030110 0 12737 000101 172540 MOV #101,@#PLKCSR ;SET IE,100KHZ AND GO BITS
5653 030116 0 00415 BR 15
5654 030120 0 32737 001000 000764 1051 BIT #LKOPT,@#OPT,CP ;BRANCH IF 'L' CLOCK NOT AVAIL
5655 030126 0 01560 ARBEX ;SKIP PRIORITY ARBITRATION TEST
5656 ;BELOW IF NO KW11-L OR KW11-P
5657 030130 0 12737 000100 177546 MOV #100,@#LKS ;SET IE BIT
5658 030136 0 12767 177546 000104 MOV #LKS,65
5659 030144 0 12767 000240 000174 MOV #NOP,95
5660
5661 ;ROUTINE TO CHECK PRIORITY ARBITRATION LOGIC
5662 ;THE BELOW TEST WILL INHIBIT INTERRUPTS ON LEVEL 6 AND ABOVE (LOCKING
5663 ;OUT THE LINE CLOCK) AND THEN SET UP THE TTY TO INTERRUPT, NEXT THE
5664 ;PRIORITY LEVEL WILL BE SET TO 0 ALLOWING INTERRUPTS IN WHICH CASE
5665 ;THE LINE CLOCK (AT LEVEL 6) SHOULD INTERRUPT BEFORE THE TTY (AT LEVEL 4).
5666 030152 132737 000020 177776 151 BITB #20,@#PSW
5667 030160 001143 BNE ARBEX ;EXIT TEST IF 'T' BIT SET
5668 030162 032737 000100 177570 BIT #100,@#SWR ;BRANCH IF USER HAS DESLECTED

```

```

DC0KCD TELETYPE & CLOCK TESTS
5669 030170 001137 BNE ARBEX ;CLOCKS
5670 030172 032737 000100 177564 251 BIT #100,@#TPS ;WAIT FOR TTY TO BE NOT
5671 030200 001374 BNE 25 ;BUSY
5672 030202 112737 000300 177776 MOVB #300,@#PSW ;SET PRIORITY LEVEL 6
5673 030210 152737 000100 177564 351 BITB #100,@#TPS ;SET IE BIT
5674 030216 100374 BPL 35 ;AND WAIT FOR EADY
5675 030220 013767 000064 000210 MOV @#TRVEC,@#TPVEC ;SAVE TTY VECTOR
5676 030226 012737 030312 000064 MOV #75,@#TPVEC ;SET TTY VECTOR
5677 030234 005027 CLR (PC)* ;CLEAR CHECK WORD
5678 030236 000000 451 ;WORD 0
5679 030240 000240 NOP
5680 030242 000240 NOP
5681 030244 000240 NOP
5682 030246 113700 551 MOVB @#(PC)+,R0 ;GET CLOCK STATUS & BRANCH IF READY
5683 030250 172540 651 ;WORD PLKCSR ;CONTAINS ADDRESS OF L OR P CLOCK STATUS REG.
5684 030252 100375 BPL 55
5685 030254 000240 NOP ;AT THIS TIME BOTH THE CLOCK
5686 ;AND THE TTY ARE READY TO INT
5687 030256 012737 030326 000100 MOV #85,@#LKVEC ;SET CLOCK VECTOR
5688 030264 013737 000100 000104 MOV @#LKVEC,@#PLKVEC
5689 030272 105037 CLR @#PSH ;SET PRIORITY LEVEL 0
5690
5691 030276 022767 000022 177732 CMP #2,45 ;CHECK THAT THE CLOCK
5692 030304 021455 BEQ ARBFIN ;& TTY INTERRUPTED IN
5693 030306 104400 HLT ;THE PROPER SEQUENCE
5694 030310 000453 RR ARBFIN
5695
5696 030312 042737 000100 177564 751 RIC #100,@#TPS ;CLEAR IE BIT
5697 030320 006367 177712 ASL 45 ;SHIFT INDICATOR
5698 030324 000002 RTI ;RETURN
5699
5700 030326 005267 177704 851 INC 45
5701 030332 012737 030350 000100 MOV #LKSRV,@#LKVEC ;SET CLOCK VECTORS
5702 030340 012737 030400 000104 MOV #PLKSRV,@#PLKVEC
5703 030346 003414 BR PLKSRV ;FINISH SERVICE (NOTE: CONTAINS NOP IF NO P CLOC
5704
5705
5706 030350 005267 150420 LKSRV INC LTICKS ;INCREMENT CLOCK TICK COUNT
5707 030354 012737 000100 177546 MOV #100,@#LKS ;CLEAR READY
5708 030362 032737 000100 177570 RTI #100,@#SWR ;BRANCH IF USER DESIRES TO
5709 030370 001402 REQ 15 ;KEEP CLOCK ENABLED
5710 030372 005037 177546 CLR @#LKS ;DISABLE CLOCK
5711 030376 000002 151 RTI ;RETURN
5712
5713 ;KW11-P INTERRUPT SERVICE
5714 030400 005267 150372 PLKSRV INC PTICKS
5715 030404 012737 000100 172542 MOV #100,@#PLKCSB
5716 030412 012737 000101 172540 MOV #101,@#PLKCSR ;RE-ENABLE KW11-P
5717 030420 032737 000120 177570 RTI #100,@#SWR
5718 030426 001402 REQ 15
5719 030430 005037 172540 CLR @#PLKCSR
5720 030434 000002 151 RTI
5721
5722 030436 000000 ;TPVEC ;WORD 0

```

```

5723 030440 013737 00436 000064 ARBFIN; MOV    @#,TPVEC,@#TPVEC    ;RESTORE TTY VECTOR
5724 030446 042737 00100 177564      RIC    #100,@#TPS
5725 030454 012737 002350 000100      MOV    #LKSrv,@#LKVEC    ;ISFT CLOCK VECTORS
5726 030462 012737 030400 000104      MOV    @PLKSRV,@#PLKVEC
5727 030470 104000      ARBEX; SCDFE
5728
5729
5730 030472 032737 001000 000764      ;TURN ON KW11=CLOCK IF BOTH ARE AVAILABLE
      BIT    #LKOPT,@#OPT,CP    ;BRANCH IF NOT AVAIL
5731 030500 001411      BEQ    1$
5732 030502 012737 030350 000100      MOV    #LKSrv,@#LKVEC    ;SET VECTOR
5733 030510 012737 000300 000102      MOV    #300,@#LKVEC*2    ;AND PRIORITY LEVEL 6 ON INT,
5734 030516 052737 000100 177546      BIS    #100,@#LKS        ;SET IE BIT
5735 030524
    1$1
    
```

```

5736      ;HIGH SPEED READER TESTS
5737      ;TO RUN TEST LOAD PAPER TAPE TEST LOOP (A BINARY COUNT PATTERN) INTO
5738      ;READER AND ENABLE READER,
5739
5740      ;ROUTINE TO SYNCHRONIZE READER, THE ROUTINE READS DATA UNTIL A 0 CHAR=
5741      ;ACTER IS FOUND AND THEN SWITCHES THE INTERRUPT TO THE READER SERVICE
5742      ;ROUTINE BELOW (PRST)
5743 030524 032737 002001 000766      BIT    #PROPT,@#OPTIONS    ;CHECK IF READER IS AVAILABLE
5744 030532 001506      BEQ    PREXIT
5745 030534 012737 030572 000070      MOV    @PRSTR,@#PRVEC    ;SET INTERRUPT VECTOR
5746 030542 012737 000400 000072      MOV    #400,@#PRVEC*2    ;SET PRIORITY LEVEL 4
5747 030550 112737 000001 000770      MOV    #1,@#PRDAT        ;SET FIRST DATA CHAR
5748 030556 105037 000771      CLR    @#PRSYNC        ;INITIALIZE SYNC COUNT
5749 030562 012737 000101 177550      MOV    #100,@#PRS        ;SET READER ENABLE & IE
5750 030570 000467      BR    PREXIT            ;CONTINUE TEST
5751
5752 030572 032737 100200 177550 PRSTR; BIT    #100200,@#PRS    ;BRANCH IF READY & NO ERROR
5753 030600 003002      BGT    1$
5754 030602 104400      HLT
5755 030604 000461      BR    PREXIT
5756
5757 030606 105237 000771      1$; INCB  @#PRSYNC        ;COUNT A MAXIMUM OF 120, NON
5758 030612 001002      BNE    2$              ;'0' CHARACTERS
5759 030614 104400      HLT                    ;ERROR! MORE THAN 120 0 CHARACTERS TO
5760 030616 000410      BR    3$              ;CHARACTERS DETECTED
5761
5762 030620 105737 177552      2$; TSTB  @#PRB          ;BRANCH IF A NON 0 CHARACTER
5763 030624 001005      BNE    3$
5764 030626 105037 000771      CLR    @#PRSYNC        ;RESET SYNC
5765 030632 012737 030646 000070      MOV    @PRST,@#PRVEC    ;SET VECTOR TO TEST
5766 030640 005237 177550      INC    @#PRS            ;SET READER ENABLE
5767 030644 000002      RTI                    ;EXIT INTERRUPT
5768
5769
5770      ;HIGH SPEED READER TEST,
5771      ;ROUTINE READS EACH CHARACTER FROM TAPE EXPECTING THE CHARACTER TO BE
5772      ;1 GREATER THAN LAST ONE READ UNTIL A ZERO CHARACTER IS READ, WHEN
5773      ;A ZERO CHARACTER IS READ THE TEST ALLOWS UP TO 120, 0 CHARACTERS TO
5774      ;BE CONSECUTIVELY READ BEFORE A '1' CHARACTER IS FOUND,
5775 030646 032737 100200 177550 PRST; BIT    #100200,@#PRS    ;BRANCH IF READY & NO ERROR
5776 030654 003003      BGT    1$
5777 030656 104400      HLT                    ;ERROR! ERROR BIT OR NOT READY
5778 030660 005746      TST    -(SP)            ;PUSH FAKE CHAR ON STACK
5779 030662 000426      BR    100$            ;EXIT VIA 100$
5780
5781 030664 113746 177552      1$; MOV    @#PRB,-(SP)        ;GET CHARACTER READ
5782 030670 001413      BEQ    2$              ;BRANCH IF A '0'
5783 030672 121637 000770      CMP    (SP),@#PRDAT      ;BRANCH IF INCORRECT DATA READ
5784 030674 001017      BNE    99$
5785 030676 105237 000770      INCB  @#PRDAT            ;SET NEXT CHARACTER
5786 030678 001015      BNE    100$           ;BRANCH IF NOT '0'
5787 030680 105237 000770      INCB  @#PRDAT            ;MAKE NEXT CHAR TO BE A '1'
5788 030682 105037 000771      CLR    @#PRSYNC        ;INITIALIZE SYNC COUNT
5789 030684 000410      BR    100$            ;EXIT VIA 100$
    
```

5790	030720	12737	000001	000770	251	CMPB	#1,0#PRDAT	ICAN ONLY READ A '0' BEFORE A '1'
5791	030726	001033				BNE	995	
5792	030730	105237	000771			INCB	0#PRSYNC	ICOUNT MAXIMUM OF 128 '0' CHARS
5793	030734	001001				BNE	1005	
5794	030736	104400				995:	HLT	IREADER DATA ERROR
5795	030740	005226				1005:	TST	IPOP DATA READ
5796	030742	005237	177550				INC	ISET READER ENABLE
5797	030746	000002					RTI	IEXIT TEST
5798								
5799	030750							

PREXIT:

5800	030750	004767	000232		ENDI	JSR	PC,STHM	I00 START RELOCATING ABOVE 28K
5801	030754	012737	009274	000004	END1:	MOV	#ERRRT,0#ERRVEC	
5802	030762	005037	177776			CLR	0#PSW	ICLEAR MODE BITS IN PSW
5803	030766	004767	151540			JSR	PC,CLRTBIT	I00 CLEAR '1' BIT IF SET
5804	030772	012700	000600			MOV	#KPTR,SP	ISET KERNEL STACK PTR
5805	030776	032737	000400	000764		BIT	#TTOPT,0#OPT,CP	I00 BRANCH IF NO CONSOLE TTY
5806	031024	001433				BEQ	15	
5807	031026	032737	000100	177564		BIT	#100,0#TPS	ICHECK IF OUTPUT DEVICE IS BUSY
5808	031014	001374				BNE	1-6	IIS AVAILABLE
5809	031016	105737	177570			TSTB	0#SHR	IDELETE END OF PASS TYPE OUT IF SW7=0
5810	031022	100024				BPL	15	I00 BRANCH IF SW7 IS DOWN
5811	031024	016702	147750			MOV	ICNT,R2	I00 GET PASS COUNT
5812	031030	004767	151636			JSR	PC,ENVDAT	I00 CONVERT TO ASCII STRING
5813	031034	012702	003076			MOV	#D16ITS+2,R2	I00 GET ASCII VALUES
5814	031040	012703	003574			MOV	#PASSES,R3	I00 MOVE THEM INTO MESSAGE
5815	031044	012704	003750			MOV	#PASSNO,R4	
5816	031050	011223				MOV	(R2),(R3)+	
5817	031052	012224				MOV	(R2),(R4)+	
5818	031054	011223				MOV	(R2),(R3)+	
5819	031056	012224				MOV	(R2),(R4)+	
5820	031060	012737	003642	030030		MOV	#SUCCESS,0#MSG	I00 PASS MESSAGE ADRS TO TELETYPE SERVICE
5821	031066	052737	000100	177564		BIS	#100,0#TPS	I00 SET IE BIT
5822	031074	005267	147700		151	INC	ICNT	
5823	031100	116700	147660			NOVB	OPT,CP,R0	I00 GET CP TYPE
5824	031104	026067	032252	147666		CMP	PASTAB(R0),ICNT	ICHECK IF END OF TEST
5825	031112	001002				BNE	25	I00 BRANCH IF NOT AT END
5826	031114	000167	000776			JMP	DONE	
5827	031120	016702	147654		251	MOV	ICNT,R2	I00 GET PASS COUNT
5828	031124	006302				ASL	R2	
5829	031126	046002	032226			BIC	CPPASS(0),R2	I00 LIMIT PASS COUNT TO 0-6
5830	031132	012737	000016	000014		MOV	#16,0#14	I00 RESTORE VECTOR FOR RTT TEST
5831	031140	005037	000016			CLR	0#16	ICLEAR T BIT TRAP ADDRESS
5832	031144	113737	001151	001150		NOVB	0#ITCNT=1,0#ITCNT	I00 RESET ITERATION COUNT
5833	031152	016216	032206			MOV	PSWTAB(2),(SP)	I00 PUSH NEXT PASS PSW ON STACK
5834	031156	032716	000020			BIT	#20,(SP)	I00 WILL 'T' BIT BE SET ON NEXT PASS?
5835	031162	001406				BEQ	35	I00 BRANCH IF NOT
5836	031164	112737	000001	001150		NOVB	#1,0#ITCNT	I00 SET ITERATION COUNT = 1 FOR 'T' BIT
5837	031172	016737	000006	000016		MOV	RTI,0#16	I00 SET 'T' BIT TRAP TO RETURN VIA 16
5838	031200	012746	005644		351	MOV	#START2,=(SP)	I00 RESART PROGRAM AT START2
5839	031204	000000			RTI1:	RTT		I00 RESTART PROGRAM AT START2 WITH NEW PSW
5840								I(FROM TABLE BELOW)
5841								

```

DC0KCD STMM ROUTINE
5842          .SBTTL STMM ROUTINE
5843          JROUTINE TO SET UP MEMORY MANAGEMENT TO RELOCATE PROGRAM CODE ABOVE 28K
5844 031206 005737 002764          STMM1 TST      0#OPT,CP          ;CHECK FOR MEM MGMT OPTION
5845 031212 100401          BHI      2$          ;BRANCH IF AVAILABLE
5846 031214 000207          RTS      PC          ;RETURN
5847
5848 031216 032737 001000 177570 2$; BIT      #1000,0#SWR          ;BRANCH IF SW09 IS = 0
5849 031224 001406          BEO      3$
5850 031226 032737 010000 177570 BIT      #10000,0#SWR          ;JMP IF NO RELOCATION ABOVE 28K
5851 031234 001010          RNE      4$          ;IE SW12=0 & SW9=1
5852 031236 000167 000614          JMP      ENDM          ;EXIT
5853 031242 032737 010000 177570 3$; BIT      #10000,0#SWR          ;BRANCH IF SW12=0
5854 031250 001402          BEO      4$
5855 031252 000167 000600          JMP      ENDM
5856 031256 013727 177776          MOV      0#PSW,(PC)+          ;SAVE OLD PSW
5857 031262 000000          OLDPSW1 WORD 0
5858 031264 012737 002200 177776 MOV      #PRTY4,0#PSW          ;SET LEVEL 4 & KERNEL MODE
5859 031272 004767 151234          JSR      PC,CLRTBIT          ;CO CLEAR 'I' BIT IF SET
5860 031276 012700 077406          MOV      #77406,R0
5861 031302 010037 172300          MOV      R0,0#KIPDR0          ;SET KIPDR0,1,6 & 7 R/W UP 4K WORDS
5862 031306 010037 172302          MOV      R0,0#KIPDR1
5863 031312 010037 172304          MOV      R0,0#KIPDR2
5864 031316 010037 172306          MOV      R0,0#KIPDR3
5865 031322 010037 172316          MOV      R0,0#KIPDR7
5866 031326 005037 172340          CLR      0#KIPAR0
5867 031332 012737 002000 172342 MOV      #200,0#KIPAR1
5868 031340 016737 000442 172344 MOV      NEXPAR,0#KIPAR2
5869 031346 013737 172344 172346 MOV      0#KIPAR2,0#KIPAR3
5870 031354 062737 002200 172346 ADD      #200,0#KIPAR3
5871 031362 012737 177600 172356 MOV      #177600,0#KIPAR7
5872 031370 005046          CLR      -(SP)
5873 031372 032737 011000 177570 BIT      #10000,0#SWR          ;
5874 031400 001006          BNE      1$
5875 031402 122737 000010 000764 CMPB     #10,0#OPT,CP
5876 031410 001002          BNE      1$
5877 031412 012716 000020          MOV      #20,(SP)
5878
5879 031416 010037 177600          1$; MOV      R0,0#UIPDR0          ;SET UP USER MEM MGMT REGS
5880 031422 010037 177622          MOV      R0,0#UIPDR1
5881 031426 010037 177616          MOV      R0,0#UIPDR7
5882 031432 016737 002350 177640 MOV      NEXPAR,0#UIPAR0
5883 031440 013737 177640 177642 MOV      0#UIPAR0,0#UIPAR1
5884 031446 062737 002000 177642 ADD      #200,0#UIPAR1
5885 031454 013737 172356 177656 MOV      0#KIPAR7,0#UIPAR7
5886
5887 031462 122737 000004 000764 CMPB     #4,0#OPT,CP          ;BRANCH IF AN 11/40
5888 031470 001424          BEO      3$
5889 031472 010037 172200          MOV      R0,0#SIPDR0          ;SET UP SUPERVISOR MEM MGMT REGS
5890 031476 010037 172202          MOV      R0,0#SIPDR1
5891 031502 010037 172216          MOV      R0,0#SIPDR7
5892 031506 016737 000274 172240 MOV      NEXPAR,0#SIPAR0
5893 031514 013737 172240 172242 MOV      0#SIPAR0,0#SIPAR1
5894 031522 062737 002000 172242 ADD      #200,0#SIPAR1
5895 031530 013737 172356 172256 MOV      0#KIPAR7,0#SIPAR7

```

```

DC0KCD STMM ROUTINE
5896 031536 011637 172516          MOV      (SP),0#SR3          ;SETUP SR3
5897
5898 031542 005726          3$; TST      (SP)+          ;POP STACK
5899 031544 000240          NOP
5900 031546 012737 000001 177572 MOV      #1,0#SR0          ;ENABLE MEM MGMT (PROG IS IN KER MODE)
5901 031554 105237 000770          INCB     0#MMON          ;SET MEM MGMT ON IND = ON
5902 031560 004767 153402          JSR      PC,LDDISP          ;LOAD DISPLAY REGISTER
5903 031564 013767 005172 000576 MOV      0#DISPLY,ENDTAG          ;AND ALSO AS LAST WORD XFERED
5904 031572 012737 032054 000004 MOV      ENDMEM,0#ERRVEC          ;SET TIME OUT TRAP VECTOR
5905 031600 012700 042000          RETRY; MOV      #40000,R2
5906 031604 005000          CLR      R0
5907
5908 031606 012704 032372          MOV      #ENDTAG+2,R4          ;DATA WILL BE RELOCATED FROM
5909 031612 010203          MOV      R2,R3          ;ADDRESS IN R0 TO ADDRESS IN R2
5910 031614 000403          ADD      R4,R3          ;GET # OF BYTES TO RELOCATE
5911 031616 010013          MOV      R0,(R3)          ;GET 'TO' ADDRESS
5912 031620 012737 005274 000004 MOV      #ERRRT,0#ERRVEC          ;FORM LAST 'TO' ADDRESS FOR RELOCATION
5913 031626 032737 000040 177570 BIT      #40,0#SWR          ;CHECK IF SUFFICIENT MEMORY AVAILABLE
5914 031634 001007          RNE      11$          ;RESTORE ERROR TRAP VECTOR
5915 031636 032737 000010 177570 BIT      #10,0#SWR          ;CHECK IF ALL DEVICES DESIRED FOR
5916 031644 001431          BEO      1$          ;RELOCATION ROUND ROBIN STYLE
5917 031646 113737 177570 000757 MOVB     0#SWR,0#DEV          ;CHECK IF A DEVICE IS SPECIFIED
5918 031654 005046          CLR      -(SP)          ;GET SELECTED DEVICE
5919 031656 013702 172344          MOV      0#KIPAR2,R2          ;CLEAR WORKING LOCATION
5920 031662 006302          ASL      R2          ;FORM ADDRESS FOR READ DATA
5921 031664 006302          ASL      R2          ;SHIFT KIPAR BITS TO FORM
5922 031666 006302          ASL      R2          ;18 BIT PHYSICAL ADDRESS
5923 031670 006302          ASL      R2          ;IN R2 AND TOP OF STACK
5924 031672 006302          ASL      R2
5925 031674 006116          ROL      (SP)
5926 031676 006302          ASL      R2
5927 031700 006116          ROL      (SP)
5928 031702 006316          ASL      (SP)          ;POSITION EA BITS AT
5929 031704 006310          ASL      (SP)          ;BIT POSITION
5930 031706 006316          ASL      (SP)          ;4 AND 5
5931 031710 006316          ASL      (SP)
5932 031712 112637 000762          MOVB     0#1712,12637          ;AND SAVE IN EABITS
5933 031716 004737 001420          JSR      PC,0#IODEV          ;GO RELOCATE DATA VIA I/O DEVICE
5934 031722 102005          BVC      10$          ;GO RELOCATE DATA VIA I/O DEVICE
5935 031724 012702 040000          MOV      #40000,R2          ;BRANCH IF NO ERRORS
5936 031730 012022          1$; MOV      (R0)+,(R2)+          ;RESTORE 'TO' ADDRESS
5937
5938 031732 003022          CMP      R3,R2          ;RELOCATE PROGRAM CODE TO ADDRESS SPEC-
5939 031734 001375          RNE      1$          ;IN R2/KIPAR2
5940 031736 010302          10$; MOV      R3,R2          ;CHECK IF AT LAST ADDRESS
5941 031740 012703 001000          MOV      #1000,R3          ;DO NOT CHECK FIRST 1000 (8) LOCATIONS
5942 031744 024042 2$; CMP      -(R0),-(R2)          ;CHECK THAT DATA WAS RELOCATED PROPERLY
5943 031746 001403          BEQ      3$
5944 031750 004737 002502          JSR      PC,0#SAVVAL          ;GO SAVE APPROPRIATE VALUES
5945 031754 104400          HLT
5946
5947 031756 020003          3$; CMP      R0,R3          ;ERROR! DATA NOT RELOCATED PROPERLY
5948 031760 001371          RNE      2$          ;R0# SOURCE/R2#DEST ADDRESS
5949 031762 162737 722012 000772 SUB      #12,0#DEVIO          ;BRANCH IF ERROR ON RELOCATION

```

```

5950 031770 000000
5951 031772 100037 000797
5952 031776 000037 000772
5953 032002 060777 000040
5954 032006 000000
5955 032010 013737 172344 172340
5956 032016 013737 172346 172342
5957
5958
5959
5960
5961
5962 032024 012706 000600
5963 032030 005037 177776
5964 032034 016740 177222
5965 032040 012746 032046
5966 032044 000002
5967
5968
5969
5970
5971 032046 000240
5972 032050 000137 005644
5973

```

```

REQ RETRY
INCB @#DEV ;STEP TO NEXT DEVICE
CLR @#DEV ;SET DEVICE IND = CP
ADD #40,(PC)+ ;STEP NEXT VALUE FOR KIPAR1
NEXPAR1 WORD 0 ;CONTAINS NEXT VALUE FOR KIPAR1
MOV @#KIPAR2,@#KIPAR0
MOV @#KIPAR3,@#KIPAR1
;*****
;PROGRAM IS NOW EXECUTING IN KERNEL MODE RELOCATED TO ADDRESS AS SPEC-
;IFIED IN KIPAR0, FOR EX, IF KIPAR0#1600 THEN PROGRAM EXECUTING AT
;ADDRESS 100000+(PC)
MOV #KPTR,SP ;SET KERNEL STACK PTR
CLR @#PSW
MOV OLDPSW,-(SP) ;RESTORE OLD PSW
MOV #15,@(SP)
RTI
;*****
;PROGRAM NOW EXECUTING IN MODE AS DETERMINED BY PASS COUNT (SEE PASTAB
;BELOW), ADDRESS IS DETERMINED BY MODE'S PAR0 REGISTER,
;DON'T REPLACE WITH HALT IF USER/SUPER MODE
;*****
NOP
JMP @#START2 ;RESTART PROGRAM AT START2

```

```

5974
5975 032054 022626
5976 032056 005037 177572
5977 032062 122737 000004 000764
5978 032070 001402
5979 032072 005037 172516
5980 032076 002240
5981
5982
5983
5984
5985 032100 012767 001600 177700
5986 032106 105037 000770
5987 032112 000137 030754
5988
5989
5990 032116 032737 000100 177564
5991 032124 001374
5992 032126 105037 177566
5993 032132 105737 177564
5994 032136 100375
5995 032140 005000
5996 032142 162700 000001
5997 032146 001375
5998 032150 000005
5999 032152 105737 177570
6000 032156 100002
6001 032160 000004 032330
6002 032164 013702 000042
6003 032170 001404
6004 032172 004712
6005 032174 000240
6006 032176 000240
6007 032200 000240
6008 032202 000137 005600
6009
6010
6011
6012
6013
6014
6015
6016 032206 000000
6017 032210 000020
6018 032212 140000
6019 032214 140020
6020 032216 144000
6021 032220 144020
6022 032222 044020
6023 032224 044020
6024
6025
6026
6027 032226 177774

```

```

;WHEN RELOCATION ABOVE 28K IS COMPLETE PROGRAM TRAPS TO ENCMEM,
ENCMEM CMP (SP)+,(SP)+ ;POP STACK TWICE
ENDH1 CLR @#SR0 ;DISABLE MEM MGMT
CMPB #4,@#OPT,CP ;BRANCH IF 11/40
BEQ 15
CLR @#SR3
;*****
;PROGRAM NOW EXECUTING IN KERNEL MODE AT PC AS SHOWN (NO RELOCATION)
;*****
MOV #1600,NEXPAR ;RESET NEXT VALUE FOR PAR REGISTERS
CLRB @#MMON ;SET MEM MGMT ON IND = OFF
JMP @#END1 ;RETURN TO INST FOLLOWING JSR PC,LDMH
;*****
DONE1 BIT #100,@#TPS ;WAIT FOR TTY OUTPUT TO FINISH
BNE DONE
CLRB @#TPB ;TYPE NULL CHARACTER
TSTB @#TPS ;WAIT UNTIL DONE
BPL ;=4
CLR R0
SUB #1,R0
BNE 15
RESET
TSTB @#SWR ;BRANCH IF NOT TYPEOUT DESIRED
BPL 25
TYPE,ENDMSG
;*****
251 MOV @#42,R2 ;CHECK DDP/ACT11 MONITOR HOOK
BEQ DONE1
LOGICAL JSR PC,(R2) ;GO TO DDP/ACT11 MONITOR VIA #2
NOP
NOP
NOP
DONE11 JMP @#START3 ;RESTART PROGRAM
;*****
;THE BELOW TABLE REPRESENTS THE 'NEW' PSW SET BY THE PROGRAM ON
;SUCCESSIVE PASSES,
;NOTE THE BELOW TABLE MAY BE MODIFIED TO CAUSE THE PROGRAM TO RUN
;UNDER USER DEFINED PARAMETERS BY PATCHING IN THE DESIRED PASS PARAMETER
;FOR EXAMPLE TO CAUSE THE PROGRAM TO RUN WITHOUT SETTING THE 'T' BIT
;IN ALL PASSES PATCH OUT THE 'T' BIT IN THE TABLE,
PSWTAB1 000000 JALL 11 FAMILY CP'S
000020
140000 ;11/45, 11/40 ONLY
140020
144000 ;11/45 ONLY
144020
044020
;*****
;THE BELOW TABLE IS THE 'BIT MASK' USED TO DETERMINE THE INDEX VALUE
;NEEDED TO SET THE 'NEW' PSW,
CPPASS1 177774

```


6028	032230	177774	177774	
6029	032232	177770	177770	111/40
6030	032234	177760	177760	111/45
6031	032236	177760	177760	
6032				

```

6033          ;THE BELOW TABLE REPRESENTS THOSE BITS IN THE CP WHICH CAN BE SET/CLEARED
6034          ;EXCLUDING THE REGISTER SET BIT IN THE 11/45,
6035          PSHBITI 000377
6036          032240 000377          000377
6037          032242 000377          000377
6038          032244 000357          111/40
6039          032246 170357          111/45 (REGSET BIT IS CHECKED ELSEWHERE)
6040          032250 170357          170357
6041
6042          ;THE BELOW TABLE CONTAINS THE # OF PASSES REQUIRED TO COMPLETE TEST
6043          PASTABI  WORD 2
6044          032252 000002          WORD 2
6045          032254 000002          WORD 2
6046          032256 000004          WORD 4          111/40
6047          032260 000010          WORD 10         111/45
6048          032262 000010          WORD 10
6049
6050          ;MESSAGES
6051          AOPT,CP1,ASCIZ '<15><12>'OPT,CP# '
6052          MSG1:  ASCIZ '<15><12>'LOW LIMIT?'
6053          MSG2:  ASCIZ 'HIGH LIMIT?'
6054          ENDMMSG; ASCIZ '<15><12>' DCOKC DONE'
6055
6056          ;EVEN
6057          NODEVI;  ASCIZ '<15><12>'
6058          DEVNAH;  ASCIZ 'XX NOT ON BUS'<15><12>'
6059
6060          ;EVEN
6061          ENDTAGI; WORD 0
6062          ;NOTE: THE FOLLOWING CODE GETS OVERLAID WHEN THE PROGRAM IS STARTED,
6063          ;THESE ROUTINES ARE USED TO CHECK THE TYPE,HLT,AND SCOPE ROUTINES,
6064          ;CHECK TYPE ROUTINE:
6065          CHKTYPI; MOV #500,SP          ;SET STACK PTR
6066          TYPE,SUCCESS
6067          RR          CHKTYYP
6068
6069          ;CHECK HLT ROUTINE:
6070          CHKHLTI; MOV #500,SP          ;SET STACK PTR
6071          HLT
6072          RR          CHKHLTY
6073
6074          ;CHECK SCOPE ROUTINE:
6075          CHKSCPI; MOV #500,SP          ;SET STACK PTR
6076          CLR R0
6077          MOV PC,R1          ;SET SCOPE PTR
6078          MOV R0,#*DISPLAY
6079          INC R0
6080          SCOPE
6081          RR          CHKSCPY
6082
6083
6084
6085
6086

```

6087 00001 .END

AC0 =X000000	AC1 =X000001	AC2 =X000002	AC3 =X000003
AC4 =X000004	AC5 =X000005	ACB2 011574	ACB5 012404
ADCB6 013072	ADCB7 013740	ADC0 007534	ACC1 010410
ADC2 011404	ADC5 012212	ADC6 012302	ADC7 013634
ADD0 014426	ADD1 014524	ADD1A 014750	ADD1B 014766
ADD2 015354	ADD3 016122	ADD6 016464	ADD7 017126
ADRSIS 004744	ADRTAB 002226	ADPT,C 032264	ARBEX 030470
ARBFIN 030440	ASCAN 005031	ASHCL0 025120	ASHCR0 025176
ASHL0 024710	ASHL1 025502	ASHR0 025024	ASHR1 025570
ASLB1 010752	ASLB1A 011176	ASLB3 012374	ASLB4 011700
ASLB6 013054	ASLB7 014036	ASL0 007656	ASL1 010564
ASL3 012126	ASL4 011476	ASL6 012652	ASL7 013462
ASRB1 011046	ASRB1A 011062	ASRB2 011644	ASRB2A 011662
ASRB5 012334	ASRB6 013172	ASRB7 014054	ASR0 007704
ASR1 010452	ASR2 011420	ASR3 012112	ASR6 012534
ASR7 013516	A,DATA 004136	B0ADR 004144	B0DAT 005110
BELL 003640	BICB1 015142	BICB1A 015164	BIC0 014340
BIC1 014646	BIC2 015444	BIC3 016334	BIC7 017730
BINB 015664	BINB7 017476	BIN1 015322	BISB1 015130
BIS0 014316	BIS0A 014374	BIS1 014634	BIS2 019402
BIS2A 015504	BIS7 017670	BITB1 015120	BIT02 015752
BITB3 016304	BITB6 016670	BIT1 014562	BIT13 = 020000
BIT14 = 040000	BIT15 = 100000	BIT2 015470	BIT0 = 000100
BIT8 = 000400	BPTVEC= 000014	BUBADR 001364	C = 000001
CBIT 022716	CC0 007350	CC1 007364	CC2 007400
CC3 007412	CC4 007426	CHKHLT 032404	CHK9CP 032414
CHKSP 022516	CHKTYP 032372	CLRTB1 002532	CLR0 007452
CMPB1 015074	CMPB2 015736	CMPB3 016316	CMPN 022752
CMP0 014220	CMP0A 014462	CMP1 014544	CMP1A 014664
CMP2 015372	CMP7 017062	CNTRLC= 000003	CNVADR 002716
CNVDAT 002672	CNVDIG 003016	COMB1 011030	COMB1A 011210
COMB2 011556	COMB5 012312	COMB6 013124	COMB7 013754
COM0 007516	COM1 010576	COM3 012170	COM4 011306
COM6 012520	COM7 013606	CPCHK 006000	CPERR 003614
CPPASS 032226	CPUERR= 177766	CRLF 000752	CYLADR 002214
DBINB7 017472	DBIN7 016762	DDATA 016322	DDATAB 016756
DECB1 011000	DECB1A 011130	DECB2 011712	DECB5 012452
DECB6A 013224	DECB7 014022	DEC0 007576	DEC1 010370
DEC1A 010640	DEC2 011454	DEC5 012144	DEC6 012666
DEC7 013500	DEV 000757	DEVERR 000742	DEVICE 004156
DEVID 000772	DEVNAH 002350	DEVTBL 002206	DIGBUF 003072
DIC1TS 003074	DIGTAB 003554	DISPLA= 177570	DISPLY 005172
DIV0 025402	DONE 002116	DONE1 032202	DSCAN 005046
DSKADR 002124	EABITS 000762	ECHO 000740	EISOPT= 040000
EMTVEC= 000030	EMT1 021174	EMT1B 021254	EMT1C 021260
EMT1D 021272	END 000750	ENDCP 025554	ENDM 032056
ENDMEH 032054	ENDMSG 032330	ENDTAG 032070	END1 030754
ERFLAG 005322	ERMSG 005340	ERRPT 005274	ERRREG 003621
ERRPT 005320	ERRREG= 177744	ERRVEC= 000004	ERTAG 005320
EXTINS 023160	FACTOR 001004	FISOPT= 010000	FPEVEC= 000244
FPOPT = 020300	FRSTAD 001010	FRSTME 001012	GDAUR 004066
GDDAT 005075	GSTST 010170	HALT1 026454	HLT = 104400
ICNT 001000	INCB1 010714	INCB2 012006	INCB3 012344
INCB6 013236	INCB6A 013206	INCB7 014906	INCB 007610

INC1 010504	INC3 012200	INC4 011336	INCO 012630
INC7 013646	INDEV 001420	IORETR 000960	IOTYST 020732
IOTVEC= 000020	INTCNT 001150	JMP1 020212	JMP3 020262
JMP4 020320	JMPS 020364	JMP6 020404	JMP7 014442
JSR1 020446	JSR1 020530	JSR1A 020532	JSRJ 020616
JSR3A 020620	JSR4 020674	JSR4A 020676	JSR6A 020730
KDPA0= 172360	KDPA0= 172320	KIPAR0= 172340	KIPAR1= 172342
KIPAR2= 172344	KIPAR3= 172346	KIPAR4= 172350	KIPAR7= 172356
KIPDR0= 172300	KIPDR1= 172302	KIPDR2= 172304	KIPDR3= 172306
KIPDR4= 172310	KIPDR7= 172316	KJOP1= 004000	KM = 000000
KPTR = 000600	KSP = 000000	KTABRT 005244	KTAMSG 005357
KTEX 027624	KTOPT = 100000	KTPAR 027216	KTPDR 027074
KT = 027352	KW11 030040	KDDISP 005166	LKOPT = 004000
LKS = 177546	LKSRV 030350	LKVEC = 000800	LOGICA 032172
LPB = 177516	LPS = 177514	LSYEM 005504	LTICKS 000774
MARKEX 024106	HARK1 013090	HEMTBL 000930	MNON 000770
MWVEC = 000250	MOV81 013090	HOVB 014130	MOV8A 014200
MOV1 014734	MOV7 017032	HP1 020246	MRKYST 024104
MSC = 030030	MS01 032277	HS02 032314	MUL0 025264
N = 000210	NE0B1 010744	NE0B4 010744	NEC66 013142
NE0B7 014070	NE00 007636	NE01 010030	NEC2 011360
NEC5 012156	NE66 012534	NE07 013534	NEKPAR 032006
NODEV 032346	NOTFND 004771	NULL = 000000	NULLS 030034
OAERR 017756	OLDPSP 031262	OPTON 000766	OPT_CP 000764
OFLW 171536	PARCSR= 172100	PARPR 000909	PARTBL 027334
PAPVFC= 000114	PASCNT 003564	PAUSES 003574	PASSNO 003750
PASTAL 032252	PC = 000000	PDRTBL 027316	PDWN 000610
PEFLC 000761	PERET 004402	PFAIL 000606	PFVEC = 000024
PHYSPC 003633	PIRO = 177772	PIR00 026032	PIRVEC= 000024
PIR4 = 010000	PKM = 000000	PLKCSB= 172542	PLKCSR= 172544
PLKCTR= 172544	PLKOPT= 002000	PLKSRV 030400	PLKVEC= 000104
PNTRG 003756	PPB = 177556	PPOPT = 000002	PPS = 177554
PRVEC = 000074	PRS = 177552	PRDAT 000970	PREXIT 030754
PROPT = 000001	PRB = 177550	PRTRT 030572	PRSYNC 000771
PRYST 030646	PRTY2 = 000100	PRTY3 = 000140	PRTY4 000200
PRY5 = 000240	PRTY6 = 000300	PRTY7 = 000340	PRVEC = 000070
PSM = 010000	PSW = 177776	PSWB1T 032240	PSWCHK 022304
PSMTAB 032206	PTICKS 000776	PUM = 030000	PUP 000632
QV 000771	RCA = 177452	RCS = 177446	RCD4 = 177442
RCLA = 177448	RCTBL 002374	RCVEC = 000210	RCWG = 177450
RECD 004244	REG = 004000	REGADR 004222	REGS 004200
RELOC 001152	RELPC 003626	REL1 001006	REL0 000262
REL00 007200	REL1 007262	REL11 013310	REL2 013312
REL22 017306	REL3 017310	REL33 021444	REL4 021472
REL44 023112	REL5 023114	REL55 024626	REL6 024430
REL66 026570	REL7 026572	REL77 027636	RESERR 009264
RESE1 026526	RESMSG 000374	REBPSW 002946	RESIPS 002556
RESTRP 022132	RESVEC= 000010	RETPC 001300	RETRSW 002540
RETRY 031000	RFCHA = 177464	RFD4E = 177470	RFDAR = 177466
RFDCS = 177460	RFYBL 002330	RFVEC = 000204	RFWC = 177462
RHCS2 = 000000	RKBA = 177410	RKCS = 177404	RKDA = 177412
RKOS = 177400	RKER = 177402	RKTBL 002306	RKVEC = 000220
RKWC = 177426	ROLB1 010726	ROLB2 011630	ROLB3 012420
ROLB6 013156	ROLB6A 013232	ROLB7 014106	ROLB 007670

ROL1 010424	ROL1A 010436	ROL3 012226	ROL4 011434
ROL6 012504	ROL7 013664	ROMB1 011014	ROMB1A 011104
RORB4 011612	RORB5 012362	ROMB6 013010	ROMB7 013772
RORE 007554	RORE1 010354	RORE4 010466	RORE2 011322
RORE5 012100	RORE6 012574	RORE7 013536	RORE3 176720
RPCA = 176722	RPCS = 176714	RPDA = 176724	RPDS = 176710
RPER = 176712	RPTBL 002352	RPVEC = 000254	RPWC = 176716
RP4AS = 176716	RP4B4 = 176704	RP4GA = 176734	RP4CS1 = 176700
RP4DST = 176706	RP4D51 = 176712	RP4ER1 = 176714	RP4LA = 176720
RP4OF = 176732	RP4YBL 002416	RP4VEC = 000254	RP4WC = 176702
RSAS = 172056	RSBA = 172044	RSCS1 = 172040	RSCS2 = 172050
RSDA = 172046	RSOS = 172052	RSER = 172054	RSLA = 172060
RSTBL 002450	RSVEC = 000204	RSMC = 172042	RT11 031204
RTT1 024172	RTT1EX 024362	RTY2 024070	RTT2A 024404
RTT2EX 024604	R0 = 000000	R1 = 000001	R10 = 000000
R11 = 000001	R2 = 000002	R13 = 000003	R14 = 000004
R15 = 000005	R3 = 000003	R3 = 000003	R4 = 000004
R5 = 000005	SAVPSW 024366	SAVVAL 002502	SBCB1 010740
SBCB3 012440	SBCB4 011724	SBCB6 013012	SBCB7 013720
SBC0 007720	SBC1 010520	SBC1A 010936	SBC9 012240
SBC6 012610	SBC7 013442	SBINB7 017402	SBIN7 016760
SCOPE = 004000	SCOPEA 001014	SDATA 010324	SDATAB 016754
SDPAR0 = 172260	SDPAR0 = 172220	SECT 000576	SIPAR0 = 172240
SIPAR1 = 172242	SIPAR4 = 172250	SIPAR6 = 172254	SIPAR7 = 172256
SIPDR0 = 172200	SIPDR1 = 172202	SIPDR4 = 172210	SIPDR6 = 172214
SIPDR7 = 172216	SLASH 000755	SLR = 177074	SM = 040000
SQB0 023724	SQB1 023712	SQB10 023666	SQB2 023740
SQB3 023752	SQB4 024004	SQB5 024006	SQB5A 024026
SQB6 024052	SQB7 024060	SQB8 024062	SQB9 023676
SP = 000000	SPCHK 022546	SPL0 025672	SR0 = 177572
SR1 = 177574	SR2 = 177576	SR3 = 172516	SSP = 000000
START 005422	START1 005532	START2 005644	START3 005600
STATUS 003600	STKLIN 026636	STKPTR = 000000	STHM 031206
SUB0 014166	SUB1 014602	SUB1A 014710	SUB1B 014724
SUB2 015420	SUB2A 015540	SUB3 010046	SUB3A 016070
SUB6 016504	SUB7 017102	SUCCESS 003642	SWAB0 007736
SWAB1 011144	SWAB2 011350	SWAB4 011770	SWAB6 013236
SWAB7 013570	SWR = 177570	SXRA 023534	SXRB 023536
SXT0 023202	SXT1 023266	SXT2 023552	SXT3 023566
SXT4 023334	SXT5 023376	SXT6 023476	SXT6A 023526
SXT7 023642	T = 000020	TBITVE = 000014	TKB = 177562
TKISR 003130	TKS = 177560	TKVEC = 000060	TPB = 177566
TPS = 177564	TPVEC = 000064	TRAPVE = 000034	TRAP1 021362
TRAP1C 021406	TRKSEC 002222	TRYVEC = 000014	TSTB1 011156
TSTB2 011746	TSTB2A 011756	TSTB6 012764	TST0 007474
TST1 010614	TST2 011276	TST6 013266	TTOPT = 000400
TYCHK 027640	TYPADR 003116	TYPDAT 003104	TYPE = 000004
UB6 = 013272	UBREAK = 177770	UB7 016970	UDPAR0 = 177660
UDPDR0 = 177620	UIPAR = 177640	UIPAR1 = 177642	UIPAR4 = 177650
UIPAR6 = 177654	UIPAR7 = 177656	UIPDR0 = 177600	UIPDR1 = 177602
UIPDR4 = 177610	UIPDR6 = 177614	UIPDR7 = 177616	UM = 140000
USP = 000000	UM6 012460	UMM7 013362	UM7 013366
V = 000002	VIRPC 003601	WAITID 001362	XDR0 023234
XOR1 023322	XOR24 023356	XOR35 023626	XOR6 023444

DCQKCD 11/40-11/4 CPU EXERCISER MACY 11 27(655) 4-SEP-74 11153 PAGE 129
DCQKCD SYMBOL TAB

XOR6A	023450	XOR6B	023452	XOR7	023656	E	000004
SFILLS	001002	SRESTR	005144	SSAVR	005124	,HLT	003212
,HAMF	000120	,PARSR	004356	,TPVEC	030436	,TYPE	002564
	= 032436						

ERRORS DETECTED: 0/5

*C,C/SOL-DCQKCD
RUN TIME: 23 41 0 SECONDS
CORE USED: 11K