SR23-3112-0
Courses:
11271
41242
52150
52152

IBM

Field Engineering Education
Supplementary Course Material

SYSTEM/360

Diagnostic Program
General Reference Manual

PREFACE

This book consists of a direct reproduction of existing material which is not normally available from the IBM Distribution Center. This edition is intended for educational purposes only, and therefore may not be at the latest engineering level. The courses using this book are 11271, 41242, 52150, and 52152.

(March 1970)

Issued to:	
Branch Office:	No:
Address:	
If this manual is mislaid, please return it to the above address.	

Address any comments concerning the contents of this publication to: IBM, Media Development Center, Dept 929, Rochester, Minnesota 55901.

F001-1

SYSTEM/360 DIAGNOSTIC PROGRAM

GENERAL REFERENCE MANUAL

JANUARY 15, 1967

DATE	01JUL66	15JAN67
EC NO	256929	256943

Printed in the U.S.A.

PREFACE

The type of material contained is general in nature in order that a minimum of updating will be required. Specific details of diagnostic monitors are located in respective DM user's guides. Operation details of diagnostic program sections are found in their respective write-ups. System/360 Diag. Prog. User's Guide, P/N 5396096, is discontinued because its contents have been revised and separated into individual documents as follows:

- System/360 Diag. Prog. General Reference Manual - P/N 5396300
- DM1 User's Guide - P/N 5396299
- DMK User's Guide - P/N 5396298
- DMA8 User's Guide - P/N 5396297
- FOFF User's Guide - P/N 5396291

The reference manual is designed to provide definitions and descriptions of the System/360 diagnostic maintenance program package for instructional purposes.

The DM user's guides provide information concerning specific diagnostic monitors and whatever is required to know for daily use at the system console.

This is the 1st revision of the previous reference manual which was released on EC 256929, dated July 1, 1966.

The major additions are:

1. Diagnostic utility programs - general description.
2. Diagnostic Engineering Memorandum #4
3. Updated UDT charts.

CONTENTS

FIGURES

INTRODUCTION

Diagnostic coverage for the System/360 is provided by a combination of hardware diagnostic aids, diagnostic monitor control programs and sectionalized diagnostic program tests of approximately 4K bytes each. These tests are normally referred to as sections.

The diagnostic programs in general are compatible. Exceptions will arise depending upon the characteristics of the unit being tested. Hardware diagnostic aids are acknowledged here only to introduce the total coverage available.

TYPES OF DIAGNOSTIC PROGRAMS

Maintenance programs are designed with three application environments in mind:

1. Design verification and checkout of prototypes.
2. Manufacturing and product testing support.
3. Field Usage:
   a. Initial installation checkout
   b. Unscheduled maintenance
   c. Scheduled maintenance
   d. Checkout of specific system components during customer operation
   e. Engineering change and RPQ verification

Functional Tests

These tests are intended to check that the engineering design of the system is compatible with the functional description given by the architectural manual.

Functional tests are used when it is desired to know whether a general system area (e.g., CPU, disk unit, etc.) is fault free. The design of these tests provides thorough detection coverage, short running time, and minimal size. However, thorough fault detection is not given at the circuit level.

Circuit Level Tests

These tests are used as fault-locating tools for particular system components. Information about any failures, when analyzed by the user, is sufficient to guide him to a relatively small area of circuitry.

Where possible, programs of this type are provided for all system areas not covered by fault locating tests.

Measurement Tests

This is a general category of special purpose input/output programs, each of which is designed to measure some specific parameter, such as the length of inter-record gaps, or to facilitate the manual adjustment of mechanical clearance, linkage , etc. .

Checking Circuit Tests

These tests are designed to use the hardware and program facilities to give a positive test of checking circuits.

Bring Up Tests

These simple tests can be used to bring the machine to the state where the basic diagnostics can be loaded and run.

Because these tests build up gradually, "a block at a time," these programs are most useful when there is more than one fault on the machine. The simplest programs of this type are tight loops of two or three instructions which are loaded using Initial Program Loading (IPL).

System Tests

These tests check out a system by simultaneously running all attached devices. Their prime objective is to test for erroneous interraction between devices and to determine which device is failing.

HARDWARE DIAGNOSTIC AIDS

Fault Locating Tests

These are circuit level diagnostics for the CPU'S. They are controlled by special hardware and do not require that a program be able to run. Special tests check the entire contents of read only storage(ROS).

ROS Diagnostic Tests

These tests are built into the read only storage(ROS) and require no other portion of the system to be operative

Progressive Scan Tests

These are channel fault locating tests for some 360/Systems. The tests are generated by a special set of programs and are applied under program control. These programs are run under a special monitor.

NOTE: For hardware diagnostics refer to the maintenance manual provided by the system CPU.

DIAGNOSTIC PROGRAM NOMENCLATURE

Most of the diagnostic programs are planned to run on all System/360 models, from Mod 30 thru Mod 75. This means that program size is limited to 4K bytes.

To utilize best the allotted storage space of 4K bytes, all common subroutines have been removed from the diagnostic programs, leaving only the actual test portions.

These subroutines such as print, interruption handling, loading, and I/O scheduling are placed in a single control program called a diagnostic monitor.

Diagnostic programs themselves are called sections, because they cannot run alone but must need to run under the control of the diagnostic monitor. Together, a diagnostic monitor and a program section make up a diagnostic program.

PROGRAM SECTION

Each program is divided into <u>Sections</u>. Each section is divided into <u>Routines</u>. Each section has a <u>Section Preface</u>. Each routine has a <u>Routine Prefix</u>. (See Fig. 1)

```
+---------------------------------------+
|     Diagnostic Monitor as loaded      |
|     by the IPL key from 3 to 24K      |
|---------------------------------------|
|     Section/s will be loaded on       |
|          4K boundaries                |
| +-----------------------------------+ |
| |          Section Preface          | |
| |             Section 1             | |
| |-----------------------------------| |
| |          Routine Prefix           | |
| |   -----------------------------   | |
| |                                   | |
| |             Routine 1             | |
| |-----------------------------------| |
| |          Routine Prefix           | |
| |   -----------------------------   | |
| |                                   | |
| |             Routine 2             | |
| |-----------------------------------| |
| |          Routine Prefix           | |
| |   -----------------------------   | |
| |                                   | |
| |             Routine n             | |
| +-----------------------------------+ |
| +-----------------------------------+ |
| |          Section Preface          | |
| |             Section n             | |
| |-----------------------------------| |
| |          Routine Prefix           | |
| |                                   | |
| |             Routine n             | |
| +-----------------------------------+ |
+---------------------------------------+
```

Figure 1: DM and Section Storage Map

Section: a logical independent block of coding, consisting of a section preface and a series of routines. A section may have prerequisite sections which must be run before it will operate correctly.

Section Preface: a common communication area between the diagnostic section and the diagnostic monitor. (See Fig. 2)

Routine: That part of the section containing the routine prefix and the actual test.

Routine Prefix: serves as a linkage between the routine and the diagnostic monitor.

The Section Preface

The section preface is used as a common communication area between the diagnostic section and the DM. The section preface is divided into fields which contain:

- section identification
- control bits
- interruption return addresses
- sense switches
- other reference information for both program section and DM use

The first byte of a section preface is the first byte of that diagnostic program section. All sections load into a fixed area of storage. The starting address is set to the first 4K above a diagnostic monitor. The diagnostic monitor may relocate program sections before they are executed.

The basic section preface length is 128 bytes. If the diagnostic requires unit definition table(UDT) assignments, the section preface length is increased by 4 or 8 bytes (according to the length of each entry) for each unit in the section preface unit table (SPUT). Also if a program is using program generated data, the section preface length is increased by 8 bytes if one operand is being generated and by 16 bytes if two operands are being generated.

Figure 2 shows the section preface in byte form, followed by explanations of each field. Most of the information provided is used by the programmer for writing his particular program section, but is presented here for reference for the user of the section.

Byte Hex	Byte Dec	0	1	2	3	4	5	6	7
0	0	Program and Section Number			Active RR	Sense Switches			
8	8	Reserved		PGM Intrpt Cnd. Msk		Flags *	SPUTcount	Flags **	Flags *
10	16	Initial Program Status Word							
18	24	External Old Program Status Word							
20	32	Supervisor Call Old Program Status Word							
28	40	Program Old Program Status Word							
30	48	Machine Check Old Program Status Word							
38	56	Input-Output Old Program Status Word							
40	64	Channel Status Word							
48	72	Command Address Word				RR Cycled		Extend PSW Extern	
50	80	Extnd old PSW SVC		Extend old PSW Prog		Extend old PSW MCK		Extend old PSW I/	
58	88	External Return Program Status Word							
60	96	Supervisor Call Return Program Status Word							
68	104	Program Return Program Status Word							
70	112	Machine Check Return Program Status Word							
78	120	Input-Output Return Program Status Word							
80	128	GR0 Section Register Dump Area - 64 Bytes				GR1			
88	136	GR2				GR3			
90	144	GR4				GR5			
98	152	GR6				GR7			
A0	160	GR8				GR9			
A8	168	GRA				GRB			
B0	176	GRC				GRD			
B8	184	GRE				GRF			
C0	192	Extended Control Reg(ECR) 4				ECR 5			
C8	200	ECR 6				Active Routine Prefix Address			
D0	208	Current PSW - Loaded by DM to return control to section							
D8	216	Active routine PSW - last PSW loaded by DM to start a routine							
E0	224	Section Preface Unit Table(SPUT)-all entries 4 or 8 bytes in length							
224+n		Operand 1(optional) Operand 2(optional)							

Figure 2: Section Preface Table

A description of each field in the section preface follows.

Bytes 0 - 2, program section number: contains the program identification number in hexadecimal digits.

Byte 3, current routine number: contains the number of the routine which is in operation. The section programmer codes this byte as zero. The DM updates it for each routine.

Bytes 4 - 7, section sense switches: simulated by 32 bits which are set or cleared by a CE input message. Bits 0-15 are set by DM through the use of the CE console buffer language. DM does not use, alter, or refer to these bits except as specified by input request. They may be specified for any purpose defined by the programmers e.g. stay in a tight loop for scoping.

Bytes 8 and 9: reserved for expansion.

Bytes 10 - 11, program interruption condition mask: used with the interruption (byte 15) flags to specify a particular interruption that causes DM to return to a point in the diagnostic section given by the programmer in the return PSW. The programmer places the interruption code he expects in this field. This field is examined by DM only when the program interruption flag bits(byte 15, bits 2 and 3) are set to 01.

* Byte 12, section DM flags: these flags together with byte 14 are set by the section to convey information to the DM. They are defined as follows:

Bit 0-4	Reserved for future use
5 = 0	SPUT coded as 4 bytes
= 1	SPUT coded as 8 bytes
6-7= 00	Program runs in standard PSW mode only. Note : Only those I-O devices attached to channels 0 through 6 are assigned to sections operating in this mode. If the Floating Channel feature is attached and activated, devices (on the 'floated channel') assigned to sections in this mode have the channel addresses modified to 0 through 6 before the assignment is made.
= 01	Illegal.
= 10	Program runs in extended PSW mode only. (All section preface PSW's must be assembled in extended PSW format.) Note : Any I/O devices attached to channels 0 through 63 can be assigned to sections operating in this mode.
= 11	Program can run in either PSW mode. All section preface PSW's must be assembled in standard PSW format. DM initiates each routine of the section in the standard PSW mode. Note : Any I/O devices attached to channels 0 through 63 can be assigned to sections operating with flags 6 and 7 equal to 11.

Byte 13, number of section preface unit table entries: this field gives the number of entries in the section preface unit table (SPUT). It is coded by the programmer and is used by the DM for assignment of I/O devices to the section at run time.

** Byte 14, flags set by the section: (Fig. 3) this field together with byte 12 are set by the section to convey information to the DM. They are defined as follows:

Bit 0	1	2	3	4	5	6	7
EXCLUS CPU	STOR PROT	EXCLUS CPU OVRIDE	RETURN ON CE TRMTN	DSCTNU UDT ASSGN			RETURN UNASGN I/O

Figure 3: Byte 14, Section Preface

Bit 0, exclusive CPU flag: this flag is used to indicate that a particular section cannot run in a multi-programming mode or in an OS Call situation. If this bit is 0 the section can run in all situations.

Bit 1, storage protection key assignment: this bit allows the section to control the DM handling of storage protection keys in the initial and return PSW's.

If bit is 0 - If storage protection key is available (byte 384 bit 3 of SRT), DM inserts the assigned key in all PSW's loaded except old PSW's.
1 - The protection key is not altered.

Bit 2, exclusive CPU override flag: this flag will allow execution of an exclusive CPU section to be executed in multiprogram mode.

Bit 3, return to section on CE termination: this flag is interrogated by the DM when it receives a request by the CE to terminate a section.

If bit is 0 - DM terminates the section, prints a T* message and proceeds to load the next section.
1 - The DM returns to the section via the external return PSW. This PSW must point to a subroutine in the section that performs the necessary housekeeping operations for proper termination. Upon completion of the housekeeping, the section must give an SVC D5. Upon receiving this call, the DM terminates the section, prints T* and proceeds to load the next section.

Bit 4, discontinue further UDT allocation flag: this bit is defined to allow sections to terminate themselves and not be reinitiated by the DM to run on additional units.

If bit is 0 - Continue further UDT allocations
1 - When the section issues a SVC D5 or SVC D6 FFFF, DM discontinues further UDT allocation and proceeds to terminate the section.

Bits 5 and 6: reserved for future use.

Bit 7, unassigned I/O interruptions: this bit tells the DM how to handle unassigned I/O interruptions.

If bit is 0 - Do not return unassigned interruptions to program section. The DM treats them as errors.
1 - Return all unassigned interruptions to program section.

*** Byte 15, interruption handling flags: these flags are set by the section programmer to indicate the manner in which the DM must handle each class of interruptions.

a. Treat all interruptions as errors.
b. Return all interruptions to the section, or
c. In the case of program interruptions, return a specific interruption to the section while treating all others as errors.

The setting of these flags is zero for most sections. This causes the DM to treat program interruptions, machine check interruptions, and external interruptions as errors. Supervisor call interruptions are handled as communication from the section to the DM and all assigned I/O interruptions are returned to the section.

The flags are defined as follows:

Bit 0, I/O flag: this flag directs how DM handles I/O interruptions.

If bit is 0 - DM returns to the section all I/O interruptions that are associated with assigned devices. Control is returned via the I/O return PSW (bytes 120-127 of section preface). The hardware I/O o PSW is stored in the section preface I/O o PSW.(bytes 56-63). The DM also places the hardware CSW into the channel status word (bytes 64-71) of the section preface.

1 - ~~The DM treats all~~ I/O interruptions as errors.

Bit 1: reserved for future use.

Bits 2 and 3, program interruption flag: these flags direct how the DM handles program interruptions.

If 00 - All program interruptions are treated as errors

If 01 - The DM compares the contents of the interruption condition mask (bytes 10 and 11 of section preface) and the interruption code of the hardware PSW. If not equal, DM treats the program interruption as an error. If equal, DM returns control to the section via the program return PSW (bytes 104-111 of the section preface) and stores the hardware program old PSW in the section preface program old PSW.

If 10 - All program interruptions are returned via program return PSW (bytes 104-111 of the section preface) and the hardware program old PSW is stored in the section preface program old PSW.

If 11 - Illegal; all program interruptions are treated as errors.

Bit 4, machine check: this flag directs how DM handles an internal machine check interruption.

If bit is 0 - DM treats all internal machine checks as errors.

1 - DM returns all internal machine check interruptions to the section via machine check return PSW (bytes 112-119 of the section preface). The hardware machine check old PSW is stored in the section preface machine check old PSW. DM disables hardware machine check interruptions from the time the hardware machine check new PSW is loaded, until the time the section machine check return PSW is loaded.

Bit 5, supervisor call flag: this flag controls DM's handling of supervisor call interruptions.

If bit is 0 - DM handles SVC interruptions as communication from the section to DM. All undefined codes are treated as errors.
1 - DM returns the SVC interruption to the section via the SVC return PSW (bytes 96-103 of section preface). The hardware PSW is stored in the section preface supervisor call old PSW (bytes 32-39).

Bit 6, external machine check flag: this field directs how DM handles machine check interruptions.

If bit is 0 - DM treats all external machine check interruptions as errors. (Log-out not included with message).
1 - DM returns all external machine check interruptions to the section via the machine check return PSW (bytes 112-119 of the section preface). The hardware machine check old PSW is stored in the section preface machine check old PSW (bytes 48-55).

Bit 7, external interruption flag: this flag indicates how DM handles external interruptions. This handling applies to all external interruptions except the one caused by pressing the interruption key (on the CPU console). This is reserved for use by DM.

If bit is 0 - DM treats all external interruptions as errors.
1 - The external interruptions are returned to the section via the external return PSW (bytes 88-95 of the section preface). The hardware external interruption old PSW is stored in the section preface external interruption old PSW (bytes 24-31).

Byte 16 - 23, initial program status word: this field indicates the initial running state of each routine in the section and specifies the address of the first routine prefix. The format for coding this PSW is determined by the contents of bits 6 and 7 of byte 12 (section DM flags) of the section preface.

Bits 6&7 = 00 The PSW is assembled in standard PSW format.
= 01 Illegal
= 10 The PSW is assembled in extended PSW format.
= 11 The PSW is assembled in Standard PSW format.

Since DMA8 and DMA4 run only in standard PSW mode, it will assemble the PSW in the standard PSW format only.

The storage protection bit (byte 14, bit 0 of the section preface) directs the setting of the protection key field in the PSW. This field must be normally coded with zeros to insure proper operation of this PSW when the storage protection key option is not available.

Byte 24 - 63, old program status word: DM stores a copy of the hardware old PSW into the corresponding section preface location for all interruptions that are assigned to a program section. These PSW's may then be referenced by the section instead of looking in the hardware old PSW's.

Bytes 64 - 71, channel status word (CSW): this field contains the last channel status word that DM returned to the section on an I/O interruption assigned to the section.

Bytes 72 - 75, command address word (CAW): this field contains the absolute address of the command word to be used when the next SIO is issued in problem state.

Byte 76, routine to be cycled: this field contains the number of the routine within the program section that DM is to cycle. It is set by one of three ways:

1. Manually, directly through the console keys.
2. Through the use of the CE console buffer, or
3. By entering a 'C' message through the typewriter.

Byte 77, reserved for future use:

Bytes 78 - 87, extended PSW interruption codes: these fields contain the interruption code that is stored in the hardware PSW when the CPU is operating in extended PSW mode. Not applicable to DMA4, which keeps the active routine prefix here.

Bytes 88 - 127, return program status word: these fields are used by the DM when the corresponding interruption occurs and is to be returned to the section. The same rule for coding these PSW's apply as those stated for the initial PSW.

Bytes 128 - 191, general register storage area: this is a 64 byte register save area used for storing the register contents pertaining to that section when an exit is made to DM on an interruption. Since DM does not use floating point registers, these registers remain unchanged and are not saved in the save area. DM will restore the general registers prior to returning control to the section.

Bytes 192 - 203: reserved for DM use.

Bytes 204 - 207, pointer to the active routine prefix: this field contains the absolute address of the currently active routine prefix. This field is updated by the DM each time a new routine is to be run. Not used by DMA4.

Bytes 208 - 215, current PSW: this field contains the last PSW used by the DM (LPSW) to return control to the section. Not used by DMA4.

Bytes 216 - 223, active routine PSW: this field contains the last PSW loaded by the DM to initate running a routine. Not used by DMA4.

Bytes 224 - 224+n, section preface unit table(SPUT): this is an optional field of variable length (depending on the number of entries). The format of each SPUT entry is explained below.

Bytes 224+n - 224+n+8 operand 1 (optional): this field contains 8 bytes of information which is used as the first operand by routines that reference it in the routine prefix.

Bytes 224+n+8 - 224+n+16, operand 2 (optional): this field contains 8 bytes of information which is used as the second operand by routines that reference it in the routine prefix. If a routine requires one operand, this field is omitted.

Both, Operand 1 and Operand 2, are used as line 4 of the SVC D1, D2, and D3 error messages. For SVC D1 and D2 these fields contain hex data. For SVC D3 these fields contain EBCDIC information.

Section Preface Unit Table (SPUT) (Figs. 4 and 5)

This table is supplied in every diagnostic section that requires UDT assignments. It supplies an entry for every UDT unit that the section is to test or require. This table is used by the DM to relate the symbolic unit types specified by the program section to actual unit addresses specified in the UDT. The length of each entry is either 4 or 8 bytes, depending on the setting of bit 5, byte 12 of the section preface. The number of entries must be equal to the count specified in byte 13 of the section preface.

The fields in each entry are defined as follows.

```
Bit  0      0 0     1 1 1 1            3
     0      7 8     5 6 7 8            1
    ┌──────┬───────┬─┬─┬──────────────┐
    │SYMB. │I/O    │R│A│   CHANNEL    │
    │UNIT  │OPTNS. │E│S│   AND UNIT   │
    │TYPE  │       │Q│N│   ADDRESS    │
    └──────┴───────┴─┴─┴──────────────┘
```

Figure 4: 4 byte SPUT Entry

```
Bit  0      0 0     1 1 1 1            3 3      3 4     4 4     5 5    6
     0      7 8     5 6 7 8            1 2      9 0     7 8     5 6    3
    ┌──────┬───────┬─┬─┬──────────────┬───────┬───────┬───────┬──────┐
    │SYMB. │I/O    │R│A│  CHANNEL     │DM     │       │FEATURE       │
    │UNIT  │OPTNS  │E│S│    UNIT      │CONTRL │       DEFINITION     │
    │TYPE. │       │Q│N│  ADDRESS     │       │       │       │      │
    └──────┴───────┴─┴─┴──────────────┴───────┴───────┴───────┴──────┘
```

Figure 5: 8 byte SPUT Entry

<u>Bits 0 - 7, symbolic unit type:</u> this field contains a pseudo identification supplied by the programmer for a given unit. The DM associates this unit type to a physical unit and supplies the actual address of that unit. Codes 00 and 01 are not used.

<u>Bits 8 - 15, I/O options:</u> this character indicates special features or options related to the entry in the SPUT. The field may have a different meaning for each type of device. The I/O options that apply to the assigned unit address are placed in this field by the DM at the time of UDT allocation to the section preface.

<u>Bit 16, required flag:</u> this bit, when set to a 1, tells the DM that this entry in the SPUT requires a UDT allocation in order for the section to be run. If the DM cannot assign a UDT entry (such unit type not in UDT), an RNV message is printed and the section is not executed. If the bit is set to 0, the unit can be tested but it is not required.

<u>Bit 17, assigned flag:</u> this flag is used to indicate that the DM has assigned a unit address to the diagnostic section. The DM sets this flag to a 1 when it assigns a unit address to the SPUT entry.

<u>Bits 18 - 31, channel and unit address:</u> this field contains the actual channel and unit address for the device. Supplied by the DM at the time of UDT allocation.

<u>Bits 32 - 39, DM control byte (8 byte format only):</u> must be coded as zeros. Bit 39 has been reserved for future expansion of unit types.

<u>Bits 40 - 63, feature definition field (8 byte format only):</u> this field is an extension of the I/O option field.

THE ROUTINE PREFIX.

Each routine contains a 4 byte prefix which is used by DM to pick up identification for error purposes and to pick up the address of the next routine for sequencing purposes. Each prefix starts on a word boundary.

A description of each field in the routine prefix follows.

```
Bit  0        0 0 0 1 1    1 1       2 2      3
     0        7 8 9 0 1    5 6       3 4      1
     ┌────────┬─┬─┬─┬──────┬─────────┬────────┐
     |ROUTINE |O|O|S|      |NEXT     |        |
     |Number  |P|P|K|      |   ROUTINE        |
     |        |1|2|P|      |      ADDRESS     |
     └────────┴─┴─┴─┴──────┴─────────┴────────┘
```

Figure 6: Routine Prefix

Bits 0 - 7, routine number: this field contains the routine identification number. Each routine has a unique number. This number is written as a 2 digit hexadecimal value. 00 must not be used to designate a routine number.

Bit 8, operand 1 field present: this flag indicates that the section preface contains the Operand 1 field and that this routine is using this field. If the flag is 0, this routine does not use this field. If the flag is 1, the routine is using this field for storage of program generated data. It should be noted that the flag should be zero if the routine is ~~not using this field~~ even though the field may be present in the section preface.

Bit 9, operand 2 field present: this flag indicates that the section also contains the Operand 2 field. The flag definition and section considerations are the same as for the Operand 1 flag. When used, the Operand 1 flag must also be set.

Bit 10, routine bypass flag: if this flag is set to 1 it indicates that the DM should skip this routine if DM is operating in multi-program mode.

Bits 11 - 15: not used.

Bits 16 - 31, address of next routine: this field contains the low order 16 bits of the address, relative to the section origin, of the prefix of the next routine. If this is the last routine, the field will contain the four hexadecimal digits, FFFF.

If the address of the next routine is FFFE, it indicates that the program section wants DM to load an overlay section that is on the loader source, and then reinitiate program execution via the initial PSW.

PROGRAM IDENTIFICATION SCHEME

Program sections are indentified by a 5 digit hexadecimal number $P_1P_2P_3S_1S$, where:

P_1 indicates the system model. Some programs run on all systems and a P_1 of F identifies them. Programs that run on some but not all models use a P_1 of E.

$P_2P_3S_1$= Machine area and/or device broken into blocks as:

000-0FF	Assigned by the CE to special diagnostics or for selection of a preferred set of diagnostics.
100-38F	CPU
400-4FF	Storage
500-5FF	Tape
600-6FF	etc. through FFF

S is the release level.

SAMPLE PROGRAM ID'S

32061 Model 30 CPU test at level 1.
E4412 Channel test, level 2. See program write-up for system models.
F5010 Test at level 0 for all 2400 tape drives for all model CPU.

PROGRAM IDENTIFICATION NUMBER CATEGORIES

The $P_2P_3S_1$ portion of the program identification is used as a search number by the diagnostic control programs. The user should become familiar with the search numbers for the programs he needs to run on his system. A general breakdown of search numbers by device or category follows: (additional devices are intermixed to utilize unused numbers)

$P_1P_2P_3S_1S$ = Program identification number
$P_2P_3S_1$ = Program search number

P_1 - System Identification

0	-	Special System (non-System/360)
1	-	Reserved
2	-	Model 20
3	-	Model 30
4	-	Model 40 or 44
5	-	Model 50
6	-	Model 65 or 67
7	-	Model 75
8	-	Reserved for expansion
9	-	Model 95
A	-	Reserved for expansion
B	-	Reserved for expansion
C	-	Reserved for expansion
D	-	FAA-9020 system
E	-	Applicable to some but not all systems
F	-	Applicable to all systems

$P_2P_3S_1$ - Machine Area and Device Breakdown

000-0FF Use by DM's, IPL, FLT, and special programs.

CPU

100-15F	Standard Instruction Set 1 (all models)
160-18F	Standard Instruction Set 2 (all models)
190-1DF	Floating Point Instructions (all models)
1E0-1FF	Decimal Instructions (all models)
200-25F	Standard Instruction Set 1 (specific model)
260-28F	Standard Instruction Set 2 (specific model)
290-2DF	Floating Point Instructions (specific model)
2E0-2FF	Decimal Instructions (specific model)
300-33F	CPU expansion (all model coverage)
340-37F	CPU expansion (specific model)
380-38F	Timer (all models)

STORAGE

390-39F	Standard Main Storage (model oriented)
3A0-3AF	Improved Main Storage (model oriented)
3B0-3BF	Main Storage Expansion (model oriented)
3C0-3C3	Local Storage (mod. 50, 60, 62, 70)
3C4-3C8	Bump (Mod. 50, 60, 62, 70)
3C9-3CF	Storage Protect (all models)
3C0-3C9	Model 40 local store
3D0-3DF	Shared Main Storage (model oriented)
3E0-3EF	Large capacity storage (model oriented)
3F0-3FF	Shared large capacity storage (model oriented)

CHANNEL

400-41F	Channels (all models)
420-43F	MPX Chan (model oriented)
440-45F	Sel Chan (model oriented)
460-47F	Storage Channel
480-4BF	2909 Channel
4C0-4CF	Channel Functional
4D0-4DF	Channel to Channel
4E0-4FF	Read write Direct

TAPE

500-52F	2400 Tape
530-53F	Hypertape
540-54F	Tape Switch 2816/Hypertape
550-5FF	Tape Expansion

FILE

600-61F	2311 File
620-63F	2302 File
640-65F	2321 File
660-67F	2303 File
680-68F	Dual Channel
690-69F	2301 Drum
6A0-6BF	2314 File
6C0-6DF	2318 File
6E0-6EF	2310(13RK) File
6F0-6FF	File Expansion

GRAPHICS AND DISPLAY

700-70F	Low Cost Display
710-71F	XY Plotter
720-72F	Administrative Processor Console
730-73F	Cypress
740-74F	1015 Inquiry Display terminal
750-75F	2250-1 Display Console - Unbuffered
760-76F	2250-1 Display Console - Buffered
770-78F	2250-2 Display Console
790-7CF	2280/81/82 Alpine Scanner/Recorder
7D0-7FF	Expansion

UNIT RECORD

800-80F	2821
810-81F	2540 Reader
820-82F	2540 Punch
830-83F	1403 Printer
840-84F	22XX
850-85F	1404
860-86F	2821 Two Channel Switch
870-87F	1442 Reader/Punch
~~880-88F~~	~~1443 Printer~~
890-89F	1445
8A0-8BF	Expansion
8C0-8CF	2501
8D0-8DF	2520
8E0-8FF	Expansion
900-90F	1050/1052
910-91F	Expansion
920-92F	2671 Paper Tape Reader
930-93F	2671 Paper Tape Punch
940-97F	Expansion

PROCESS CONTROL

980-98F	1827 Data Control
990-9FF	Expansion

PAPER DOCUMENT

A00-A1F	1419
A20-A3F	1418
A40-A5F	1412
A60-A7F	1428
A80-A8F	1285
A90-A9F	1231
AA0-AFF	Expansion

COMMUNICATION

B00-B0F	2701-SSYA
B10-B1F	2701-IBM Terminal Adapter 1 (SSA1)
B20-B2F	2701- Telegraph Adapter 1 (SSA2)
B30-B3F	2701-Telegraph Adapter 2 (SSA3)
B40-B4F	2701-IBM Terminal Adapter 2 (SSA4)
B50-B5F	2 Processor Switch
B60-B6F	2701-WTC Telegraph Adapter (SSA6)
B70-B7F	2701-Contact Sense Adapter
B80-B8F	2701-Contact Operate Adapter
B90-B9F	2701-Parallel Data Adapter
BA0-BAF	2701-Synchronous Data Adapter
BB0-BBF	2701-Comet
BC0-BCF	2701-IBM Terminal Adapter 3
BD0-BDF	2701- Binary Synchronous Adapter
BE0-BEF	2701-PDSA-1
BF0-BFF	2701- Expansion

COMMUNICATIONS CONTROL UNITS

C00-C4F	2701 Expansion
C50-C7F	Reserved
C80-C8F	2702
C90-C9F	7770
CA0-CAF	7772
CB0-CBF	2703
CC0-CCF	Reserved
CD0-CFF	Expansion

COMMUNICATIONS TERMINAL DEVICES

D00-D0F	1030
D10-D1F	1050
D20-D2F	1060
D30-D3F	10MD
D40-D4F	1070 (17LL)
D50-D5F	Model 28 TTY
D60-D6F	Model 33 TTY
D70-D7F	Model 35 TTY
D80-D8F	IBM Teletype
D90-D9F	WTC Telegraph
DA0-DAF	1016/1015 Remote Terminal
DB0-DBF	Low cost Display Remote Terminal
DC0-DCF	2250 Remote Terminal
DD0-DDF	Comet
DE0-DFF	Expansion

SPECIAL FEATURES (RPQ s)

E00-EFF (RPQ programs may also be fitted in their respective device categories)

EMULATOR DIAGNOSTICS

F00-F7F

LOG-OUT ANALYSIS AND PRINT PROGRAMS

F80-FAF

SYSTEM TESTS

FB0-FDF

UTILITY PROGRAMS

FE0-FEF	All Systems (P'OK controlled)
FF0-FFE	All Systems (Endicott controlled)

PROGRAM DOCUMENTATION

PROGRAM DESCRIPTION

The program description (write-up) is based on corporate engineering standard CES 0-0220 0, which defines minimum requirements for such documents. Since program sections run only under diagnostic monitor control, many write-ups make reference to user's guides for details related to a particular DM. In general each program write-up contains the following items.

PURPOSE. This is a concise statement about what the program is designed to do, including any limitations. The units, including any special features, that are tested by the program, should be stated.

REQUIREMENTS.

Program Requirements. This tells whether the program can run by itself, and the control programs, if any, with which it is compatible. If the program requires other programs to be present in the system, the details of such requirements are given here. Symbolic unit type and option codes must also be given here, if any.

Equipment Requirements. Identified here are the system units required by the program in order to operate. This shall be complete, with special features, such as an interval timer or the floating point option. Also included shall be a statement of the core storage size requirements and whether the program requires space in certain addresses or is relocatable.

OPERATING PROCEDURE. Standard operating procedures are a part of each diagnostic monitor user's guide and/or reference manual. Only if a standard operating procedure is not applicable then this section of the write-up shall contain a step-by-step description of what to do.

PRINT OUTS. In this section, examples of each of the print outs that can be produced by the program section can be given.

COMMENTS. Any data that are unique to the program and that do not belong to other sections of the write-up are given here.

PROGRAM LISTING

The listing (Fig. 7) is provided for all sections and monitors. Two areas of the listing provide the program logic and the post processor data. The listing contains the instructions and comments as coded by the programmer plus the machine language developed for each step.

The listing is relative to the actual core location. Core location and listing location are a multiple of 4K. This allows for an effective oring of the base register and displacement to get to the actual location.

```
001046   91 40 E 2C0     MSA3A TM  DMSW+8,X'40'          CHECK CEIN DELAY
00104A   47 80 D 056           BC  ALLOF,MSA3B           OFF, CHECK MSG. IN EXTERL
00104E   94 BF E 1C0           NI  DMSW+8,X'BF'          ON, RESET SWITCH
001052   47 F0 C A30           BC  UN,CA1                -RETURN TO CEIN
001056   91 02 E 1BB     MSA3B TM  DMSW+3,X'20'
            |                         |                      |
            |                         |                      |
            |                         |                      |
0043A2   47 10 A 3AA           BC  ALLON,IDCA2               |
0043A6   47 F0 C A30           BC  UN,CA1                NO TO CEIN DEV HNDLR
```

Figure 7: Sample Listing

POST PROCESSOR

The post processor (Fig. 8) provides a cross reference for the symbols in the listing. Only symbols used in the example of Fig. 7. are shown.

Item (1)	(2)	(3)	(4)
1	F	UN	1052, 43A6, (plus 477 times not listed here)
4	2A30	CA1	1052, 43A6
16	1B8	DMSW	1046, 104E, 1056, (plus 446 times not listed here)
1	8	ALLOF	104A, (plus 126 times)
1	1	ALLON	43A2, (plus 239 times)
4	4	IDCA2	43A2
4	1056	MSA3B	104A

Figure 8: Sample Post Processor

In reference to above, Fig. 8:

Item one(1) describes the number of bytes assigned to the symbols.
Item two (2) is the value assigned or the location at which the symbol is defined.
Item three (3) is the symbol as used by the programmer. Note that they are sorted by size and than alphabetic within each group.
Item four (4) is a list of the locations which use that symbol and are listed in ascending order.

The value of this cross reference is more readily seen when attempting to locate locations within a larger program which crosses several 4K boundaries.

OBJECT DECK LISTING - TYPE 1(HEX)

Some program listings are supplemented by a hex listing of that deck. (Fig. 9)

Th 80 card columns are printed as hexadecimal values on two sucessive lines. The 80 column printing is edited horizontally for easier reference. A blank position is left every 5 columns. Two blanks are left every 10 columns and four blanks every 40 columns.

~~EXAMPLE- 12,0,7~~ and ~~8 punched~~ in a ~~card~~ column are represented by Hex -8F-, and are printed on two lines with the 8 over the F.

NOTE: The column number and card numbers as in the example will not appear on the actual card deck listing.

```
        0  0     1     1     2     2     3     3     4     4  4     5/  /7     8
COLS    1  5     0     5     0     5     0     5     0     1  5     /  / 5     0
       --------------------------------------------------------------   -------
CD 1  |0CEC4 44444 01440 0ECFF FF44F 01040 05444 44444 44444 44/  /FFFF FFFFF
      |25240 00000 00000 14683 00000 00000 E0000 00000 04000 0/   /50830 00001
      |
CD 2  |0EEE4 01044 01440 0F300 00000 00000 40000 00011 44444/  /  44FFF FFFFF
      |23730 00000 08000 18000 00000 00001 00040 00028 0000/  /0  00830 00002
      |
CD 3  |     etc.
```

Figure 9: Sample Object Deck Listing(Type 1)

OBJECT DECK LISTING - TYPE 2(HOLLERITH)

Some program listings are supplemented by a Hollerith listing of the object deck. (Fig. 10) This method eliminates the conversion table use, enabling more rapid reconstruction of a damaged card, working directly from listing to keypunch. All released programs in the future should contain this type.

Three lines are printed. Each column is interpreted and printed as 2 or 3 Hollerith characters, one above the other. Blank columns are shown as periods.

Three passes of a card through a keypunch, one pass for each line, are required to reproduce a card.

Example: A sample taken from a listing follows:

```
PERIODS CORRESPOND TO BLANK COLUMNS

COLS 1 THROUGH 20    COLS 21 THROUGH 40    COLS 41 THROUGH 60    COLS 61 THROUGH

BRLD......A8....AAAA AAANAAA5AAOAAAOIAAOJ AAORAAOJAAOZAAO1AAO9 AAPAAAPAHAPJ501300
9         Y9    Y9Y9 8YQ98YQ 8Y8 6Y8 8Y8  8Y8 8Y8Z8Y8 8Y8 8Y8  8Y898Y889Y89
          9     9 9  999 999 999 999 999  999 999 999 999 999  999 9999 99
```

Figure 10: Sample Object Deck Listing(Type 2)

PROGRAM CARDS

This section describes the various types of cards that make up a diagnostic program card deck.

Note: CM6 means extended IBM card code.

ESD CARD

ESD - External Symbol Dictionary. The ESD card is the first card of each program deck. The ESD Card gives the name, origin, and length of the program section. This card also contains information required for the tape utility program.

Cols.	
1	12,2,9 punch
2-4	ESD(Hollerith)
5	Blank
6-8	Blank
9-10	Blank
11-12	16(12,0,1,8,9 and 12,11,1,8,9) (not used)
13-14	Blank
15-16	External Symbol Identification(not used)
17	Z for standard sections (some use an X) B for bringup sections S for SIP sections
18-22	Program name($P_1P_2P_3S_1S$, in Hollerith)
23-24	Blank
25	0(zero)
26-28	24 bit address of the first byte of section preface as assembled.(CM6)
29	Blank
30-32	Program section length in number of bytes. (CM6)
33-44	Reserved for section ID expansion
45-52	Reserved for RPQ
53	Blank
54-63	Part number(Hollerith, right justified)
64	Blank ("C" if mode of release is cards only). ("B" for binary and cards only).
65-71	E.C. number (Hollerith)
72	Blank
73-76	Section ID($P_2P_3S_1S$ in Hollerith)
77-80	Sequence number of card (Hollerith)

LDT CARD

LDT - Load Terminate Card. This card must be hand punched and is the last card of each program section.

Cols.	
1	12, 2, 9 punch
2-4	LDT
5-71	Blank
72-76	Program ID($P_1P_2P_3S_1S$ in Hollerith)
77-80	Card sequence number

TXT CARD

TXT - Text Card. There are a variable number of TXT cards with each program section. Each card contains up to 56 bytes of machine language text.

Cols.	
1	12,2,9 punch
2-4	TXT
5	Blank
6-8	24 bit address of first byte of text as assembled.
9-10	Blank
11-12	Number of bytes of text on card
13-14	Blank
15-16	External symbol identification (not used)
17-72	Text(each column,one byte)
73-76	Program ID
77-80	Sequence number of card

RLD CARD

RLD - Relocation List Dictionary. Each deck contains a variable number of RLD cards. At least 1 RLD card is required for the AL3 statement in the initial PSW of the section preface. RLD cards are used by the loader for updating a relocatable factor(1 to 4 bytes in length) defined by AL and CCW statements.

Cols.	
1	12, 2, 9 punch
2-4	RLD
5-10	Blank
11-12	Number of bytes of information in variable field(each relocation) factor requires 4 columns). fields (each relocation factor requires 4 columns)
13-16	Blank
17-20	Relocation and position leaders (not used)
21-72	Variable field. (Up to 13 relocation factors)
73-80	Program ID and sequence

DAT CARD

DAT - Data Card. This card must be hand punched and must precede each set of data records and each overlay section.

Cols.	
1	12, 2, 9 punch
2-4	DAT(in Hollerith)
5-71	Blank
72-76	Program ID($P_1P_2P_3S_1S$ in Hollerith)
77-80	Card sequence number

IPL CARD

IPL - Program Load Card. This card is the first card in a stand-alone program.

Cols.	
1-24	24 bytes for required programming
25-27	IPL(in Hollerith)
28	Blank
29-32	Program name(such as DMA8,DMC) in Hollerith
33	Blank
34-37	Blank(Field is used by Editor)
38	Number of 4K blocks that this program requires at IPL time. For example, if a DM requires 4K to operate and does housekeeping in the second 4K, it requires two 4K blocks. (Column is punched in CM6 code)
39-41	Blanks
42-44	Address(in CM6) that appears in the DMIO card. (see Note 1)
45-46	Blanks
47-48	Maximum number of entries which are allowed in the UDT punched
49	Count of cards in DMIO, and SRT field(CM6)(See note 1)
50	Count of cards in SDT field(CM6) (See note 1)
51	Count of cards in UDT field(CM6) (See note 1)
52-53	Blanks
54-63	Part number(Hollerith, right justified)
64	Blank ("C" if mode of release is cards only). ("B" for binary and cards only).
65-71	E.C. number(Hollerith)
72-76	Program ID($P_1P_2P_3S_1S$ in Hollerith)
77-80	Card sequence number(Hollerith)

Note 1 - These fields should be punched if the program is a control program and has a DMIO, SRT, and UDT. If the program is not a control program, the fields should be left blank.

END CARD

END - End Card. Last card in a stand-alone program.

Cols.	
1	12,2,9 punch
2-4	END(in Hollerith)
5-71	Blank
72-76	Program ID($P_1P_2P_3S_1S$ in Hollerith)
77-80	Card sequence number

REP CARD

<u>REP</u> - Replace card. This card provides a means for making temporary changes or corrections to a diagnostic.

Cols.	
1	12, 2, 9 punch
2-4	REP(in Hollerith)
5-6	Blank
7-12	The 24 bit address(6 hex char) of the first byte to be replaced. Use the address provided in the program listing.
13	Month *
14-15	Day
16	Year
17-72	Data - blocks of 4 digits. Each block separated by a comma. No blanks between blocks.
73-76	Section I.D.(P_2,P_3,S_1,S)
77-80	Card sequence number.

All information in columns 2-80 is punched in Hollerith code.

* Month is punched using 1 character
  Jan - Sept are 1 through 9
  Oct is alpha "o"
  Nov is "n"
  Dec is "d"

These replace cards should be inserted after the TXT cards and before the RLD cards.

DECK LAYOUT

The diagnostic tape utility program (FOFF) is designed around the diagnostic program card deck orders as described below. Any other is not recommended as results are not guaranteed.

Object Deck w/o Data cards.

ESD
TXT(s)
RLD(s)
LDT

Object Deck with Data records.

ESD
TXT(s)
RLD(s)
DAT
80 char records
LDT

Object Deck with Overlays

ESD
TXT(s)
RLD(s)
DAT
ESD
TXT(s)
RLD(s)
DAT
ESD
TXT(s)
RLD(s)
LDT

Object Deck with Data Records and Overlays

ESD
TXT(s)
RLD(s)
DAT
80 char records
DAT
ESD
TXT(s)
RLD(s)
DAT
80 char records
LDT

Object Deck for an IPL Program

IPL
Loader Cards
TXT(s)
END

NOTE: These card groups will be blocked if the diagnostic input is other than the card reader. This will be a function of the utility program.

DIAGNOSTIC MONITOR GENERAL DESCRIPTION

There are several types of diagnostic monitors, depending on the system size and the flexibility of communication desired between the DM and the program section. Presently available are the following diagnostic monitors(DM): DM1, DMA4/30, DMK, DMA8, DME, and DM44. Refer to Fig. 11. Diagnostic monitors are control programs designed to provide certain facilities for diagnostic programs, such as :

- Loading the programs.
- Defining the system configuration to each program.
- Handling interruptions for each program.
- Executing job requests via input messages.
- Printing output messages.

Detailed explanations of each diagnostic monitor are to be found in their respective user's guides:

DM1	P/N 5396299	(Poughkeepsie)
DMA4/30	P/N 840034	(Endicott)
DMK	P/N 5396298	(Poughkeepsie)
DMA8	P/N 5396297	(Poughkeepsie)
DME	P/N 5763442	(Kingston)
DM44	P/N 5820812	(Hursley, England)

	DMA4/30	DM44	DM1	DMK	DMA8	DMC	DME
Available	X	X	X	X	X	*	X
Main storage used	4K	4K	4K	4K	8K	24K	20K
Runs standard unit tests	X	X	sub set	X	X	X	X
Sequential testing	X	X	X	X	X	X	X
Multiprogram testing						X	
Multi-CPU coverage							X
Supports advanced architecture							X
Typewriter CE input					X	X	X
Appears on bills of material - models	30	44	40,50 65,75	40,50	40,50 65,75	to be delet	65,75
Load from cards	X	X	X	X	X	X	X
Load from tape	X	X	X	X	X	X	X
Load from disk			X	X	X	X	X
Stnd. config. cards			X	X	X		X
Primary application	stnd. Mod 30	stnd. Mod44	bring up	stg& emu-lator	stnd. Mod 40&up		stnd. large

Figure 11: DM Summary Chart

*** DMC is replaced by DMA8-7 and DME, and is no longer being maintained. The multiprogramming features of DMC are replaced by system tests, such as SIP and MIDAS.**

DIAGNOSTIC UTILITY PROGRAM GENERAL DESCRIPTION

Detailed explanations of the utility programs are found in their respective writeups or user's guides. The following are noted herein only to explain their purpose.

FOFF Diagnostic Tape Utility, P/N 5395610

The field engineer receives the diagnostic program package in a tape, disk, or card mode. Since diagnostic programs are sent with a unit from its place of manufacture, the field engineer may receive the program package in several modes. FOFF is a tape utility program. It is used by the field engineer to generate a single tape which contains all of the diagnostic programs for his system.

The various update functions of FOFF are:

Add a new program or programs to an existing system tape.
Alter a program on an existing system tape by changing on a byte basis the text portion of any such program.
Change or replace an entire program.
Delete a program.
Duplicate the existing system tape.
Generate a system tape from card image records.
Merge up to six system tapes into a single system tape.
Add a program at the beginning of a system tape.
Rearrange selected programs from their present position on the system tape to the area immediately following the IPL programs.
System configuration of a diagnostic monitor.

Utility functions of FOFF are:

Card to tape.
List the programs appearing on the system tape.
Punch the programs from the system tape.
Print the programs from the system tape.

FOFD Diagnostic Tape Conversion, P/N 5396294

This is a tape conversion program. Its only purpose is to convert any old master system tape generated by FOFF-0 or FOFF-1 (80 character records) into a blocked system tape, 3712(decimal) byte records, that can be processed by FOFF-4 or higher. The reason for blocking the system tape was so that FOFF could execute searching functions faster. The record size, 3712, was arbitrarily selected based on the average size of a program section.

Once, the old system tape is converted, FOFD is never used again and is to be set aside.

FOFA File Loader, P/N 5396121

This program provides the field engineer with an IPL loader program for the 2311 CE disk pack. It resides on track zero in an IPL format on a disk pack previously formatted by FOFE, the file utility progrm. FOFA is loaded in by means of setting the address of the 2311 disk file in the load unit switches and pressing the load key. Then it is used to load in a diagnostic monitor. FOFA has no control after the diagnostic monitor takes over.

FOFE Diagnostic Disk File Utility, P/N 5396099

This program provides a means of maintaining or generating his diagnostic programs on a 2311 disk pack. With it, he can generate a diagnostic disk pack or add new programs to an existing disk pack. The disk pack must have standard home addresses. The recommended program for this is FFFO, file initialization.

SETTING UP LOOPS

Let us start with basic principles.

First a 360 CPU can execute but one sequence of instructions at a time. Ordinarily, instructions are executed in the sequence in which they appear in main storage.

The sequential execution of instructions can be altered in two ways: by the execution of a branch instruction and by the interruption mechanism. The branch instruction needs no explanation. What about the interruption mechanism?

Basically, the interruption mechanism becomes active between the time when the execution of one instruction is completed or terminated, and the execution of the subsequent instruction is begun. When the interruption mechanism becomes active, it asks the question "Does there exist a condition that should cause a different stream of instructions to be executed." If the answer is no, the machine proceeds to execute the next instruction. If the answer is yes, the machine goes to a fixed location to fetch a pointer(Program Status Word, PSW) that points to a new instruction stream that corresponds to the interruption condition. The machine then proceeds to execute the new instruction stream. At the time the new PSW is fetched, the previously active PSW is stored by the machine in a fixed location to mark the point in the original instruction stream where execution was stopped.

If we represent the instruction stream of a unit diagnostic routine as

```
------
------
------
------
------
------
------
  .
  .
  .
  .
```

Then we establish a loop by inserting a branch at the desired point as

```
------ <-+
------   |
------   |
------   |Contents of loop
------   |
------   |
------   |
Branch --+
------
------
------
  .
  .
  .
```

Note that we are assuming that there is no branch internal to the loop that branches outside the loop.

So far so good. DM has not entered the picture. The difficulty comes when interruptions occur. Ordinarily five interruption new PSW's point to routines internal to DM. Thus if we are not careful, we shall break out of the loop and into DM when interruptions occur. This can be avoided in two ways: either disable the interruptions so that they cannot occur, or adjust the interruption pointers(new PSW's) so that they no longer point to DM instruction stream. But only input/output interruptions and machine-check interruptions can be disabled.

The simplest way to adjust the interruption new PSW's is to have them point to the first instruction in the loop as

```
           I/O new PSW -------------- ------ <-┐
           SVC new PSW -------↑       ------   |
       Program new PSW -------↑       ------   |
      External new PSW -------↑       ------   |
 Machine Check new PSW -------↑       ------   |
                                      Branch --┘
                                      ------
                                      ------
                                      ------
```

Observe what we have now. Barring addressing failures, no instructions other than those in the established loop will be executed. DM has been locked out! Also observe that it is not necessary to have all of the interruption new PSW's point to the same place in the loop.

Suppose, however, that we want to ignore program interruptions. That is, if instructions j and p cause program interruptions, we want to proceed to instructions j + 1 and p + 1 instead of going back to the beginning of the loop.

Recall that the program interruption old PSW will point to the instruction following the one that caused the interruption. What we want to do is to use this PSW when the program interruption occurs. This is accomplished by having the program interruption new PSW point to a LOAD PSW instruction that loads the program interruption old PSW.

Consider now what happens when instruction j causes a program interruption:

1. Execution of j completed(or terminated).
2. Interruption mechanism asks if any interruptions are pending. Program interruption is waiting to be serviced.
3. The hardware stores the current PSW, which now points to the instruction j+1, in the program interruption old PSW location.
4. The hardware fetches the program interruptions new PSW. This PSW points to the LOAD PSW instruction.
5. The LOAD PSW instruction is executed, using as its effective address, the address of the program interruption old PSW.
6. The original, updated PSW is now in force. Hence the instruction at j+1 is executed next.

```
┌─────────────────┐
│1. Store old PSW │  program
│2. Fetch new PSW │<--interruption--------j ──────────┐
└─────────────────┘                                   │
        │                          ┌─j+1 ─────        │
┌───────┴──────┐                   │                  │
│P.I. new PSW  │                   │      ─────       │
└───────┬──────┘                   │                  │
        │points to                 │      ─────       │
 LOAD PSW from P.I. old PSW location─┘                │
                                          ─────       │
                                          Branch ─────┘
                                          ─────
                                          ─────
                                          ─────
```

Notice that this method preserves the logic of the loop. The cost is that an instruction outside the loop must be executed when the interruption occurs. Clearly the other classes of interruptions could be handled in a similar way.

So we can see that all of the standard looping features that the CE has become accustomed to in earlier systems are still available in System/360.

To these have been added the ability to have the DM automatically establish loops with a minimum of manual intervention required.

If the CE knows the specific routine that he wants to have looped, he can type in the messages to load in a section and cycle a routine in that section.

DIAGNOSTIC ENGINEERING MEMORANDUMS

DIAGNOSTIC ENGINEERING MEMO #1

This memorandum calls attention to diagnostics that cannot be relocated or run from LCS, attached to a model 50, which are as follows:

- 5340 Diagnose Model 50 - Part 1
- E381 Timer
- E3C9 Storage Protect
- E424 Multiplexor channel invalid operations-routine 36 fails.
- E443 Selector channel CCW flags
- E444 Selector channel invalid operations-routine 36 fails.
- F14C Interrupt
- F521 IRG tape
- F522 IRG tape

In addition, any diagnostic program dependent on timings or other affecting factors cannot be run or relocated in LCS memory, unless specifically stated in the individual program write-up.

Note: Program level 3 of E424 and E444 corrects routine 36.

DIAGNOSTIC ENGINEERING MEMO #2

Subject: Pseudo Unit Type Codes, 24 and 25.

To avoid problems with duplicate channel/unit addresses entered in the diagnostic monitor unit definition table, it is suggested that during the initial system configurating of the DM, the user leave out the subject pseudo unit type codes and associated channel/unit addresses. These can be added later.

If these codes and associated addresses are omitted, the sections requiring these UDT entries cause the DM to print out an RNV message, meaning requested device not found, and the section is not run.(An RNV message from any section should cause you to make reference to the program write-up.)

Later, to use unit type codes 24 and 25, you need to reassign channel/unit addresses away from some other unit type.

DIAGNOSTIC ENGINEERING MEMO #3

Subject: Diagnostic Sections 43C2 and E427.

When defining a system configuration to a diagnostic monitor, direct your attention to the following:

43C2 - requires a UDT entry of:
- 200000XX - A non-existent device on the MPX channel. XX cannot be 07.
- 22000XXX - A non-existent device on an existing selector channel. XXX must be 100 for channel 1 and 200 for channel 2.

E427 - requires a UDT entry of 20000007.

DIAGNOSTIC ENGINEERING MEMO #4

Subject: FOFE Restriction with Programs
Assembled by a System/360 Assembler

The first TXT card of diagnostic programs assembled by means of a System/360 Assembler usually contains a byte count of zero. FOFE cannot handle such a card and processes it incorrectly in its file utility operations. This type of TXT card should have been removed from the object deck before the program was released. It is usually the 2nd card in the object deck and has zeros(12,0,1,8,9) in columns 11 and 12.

If you have trouble, putting a program on a disk file using FOFE, check for the above condition. This TXT card must be deleted.

FOFE is in the process of being updated and will contain a provision to sense and bypass such cards.

UNIT TYPE AND OPTIONAL FEATURE CODES

Since System/360 contains a variety of devices, there exists the task of first defining the system to the diagnostic monitor being used, so that it in turn can convey the information to a variety of program sections.

A unit definition language has been defined, so that DM and section can communicate. When the section is loaded, it asks DM if a particular device is available. DM makes its UDT table available, so that the section can test each entry in the table with its pre-coded unit type code and option bytes. If a matching entry is found, the section runs. If not, section is not executed.

Refer to individual DM user's guides for initially preparing each DM so that the system configuration data is contained within each DM. Refer to the program description of FOFF for a means of inserting this data in a DM on tape.

The following pages contain charts which show the unit type codes and options for all possible devices. Program sections that test these devices should have in their respective program write-ups, a description of the unit type code to be used, also.

The following charts, Fig. 12, show only the first two bytes of a UDT entry to be punched into the diagnostic monitor system configuration cards. The first byte is the unit code of the device represented by two hex digits. The second byte is the option field, represented by two hex digits, which are derived from the 8 bits.

The third and fourth bytes (not shown) contain a channel and unit address of the unit.

Note: these UDT charts are not intended to supercede program writeups. Detailed and up to data information is always found in the program writeups of each program in the section entitled Program Requirements.

Symbolic unit type codes, not yet assigned, are added to the UDT charts for your convenience. These will be assigned at some future date as required.

UNIT TYPE AND OPTION CODES FOR 4 BYTE ENTRY

	Unit Type		Options (Hex digit 1)				Options (Hex digit 2)			
bits	0-3	4-7	0	1	2	3	4	5	6	7
Console Interruption	1	0								
External Signal 2	1	1								
Signal 3	1	2								
Signal 4	1	3								
Signal 5	1	4								
Signal 6	1	5								
Signal 7	1	6								
	1	7								
	1	8								
Multiprocessor Unit	1	9		These option bits denote model number						
	1	A								
	1	B								
	1	C								
	1	D								
	1	E								
Needed by some sections, such as 43C2(ROS exerciser), Diagnose, and E424 (Channel)			No options, but these unit type codes mean:							
	1	F	Device not on this high speed multiplexor channel							
	2	0	Device not on this multiplexor channel (highest possible address: 7F)							
	2	1	Subchannel not on this multiplexor channel							
	2	2	Device not on this selector channel							
	2	3	Channel not on this system							
	2	4	Any device on this multiplexor channel(no duplicate adr)							
	2	5	Any device on selector channel (avoid duplicate address)							
			Note: see memo #2 and #3 on page 36.							
2701-1627 Plotter	2	6		1627 0=Mod1 1=Mod2						
2701-Data Speed	2	7								
2701-857 Telegraph	2	8								
Paper Tape RPQ Attch	2	9								

Figure 12: Unit Types and Options

	Unit Type		Options (Hex digit 1)				Options (Hex digit 2)			
bits	0-3	4-7	0	1	2	3	4	5	6	7
Any foreign or modified tape drives See program write-up	2	A	Modified 0011=Non data interruption 0100=Tape bit insert 0001=PE&non data intrpt 0010=NRZI&non data intrpt				Data converter	Foreign 001=Rd/Wr Ampex 010=Honeywell 011=RCA 7 track 100=Honeywell Rd/Wr		
2701-36 bit output and Sense Adapter	2	B								
7740 attachment	2	C								
2701-RCA EDGE	2	D								
2916	2	E								
	2	F								
MSP Multiplier Sum Processor	3	0				12 bit c'flow		000 = 16K 001 = 32K 010 = 64K		
	3	1								
2701-Autodin	3	2								
2701-223 Telereg.	3	3		223 Teleregister control unit address						
2702-mixed 81D1	3	4	8 bit meaning 00000000 = Alcoa mixed 81D1 00000001 = Western Union plan 117B mixed 81D1							
2702-Inhibit Deletion Figs & Letter	3	5								
	3	6								
	3	7								
Timer	3	8		cycles 0=60 c 1=50 c	Hi-Frq	RPQ Spec. (HRT)	300 cycle RPQ			Decremental
2909 Commun. Chan	3	9								
2909 Serial Input Ch	3	A								
2909 Parallel Input	3	B								
2909 Serial Output	3	C								
2909 Parallel Output	3	D								
2909 Half Duplex	3	E								
2909 Reserved	3	F								

	Unit Type		Options (Hex digit 1)				Options (Hex digit 2)			
bits	0-3	4-7	0	1	2	3	4	5	6	7
2846 Chan Ctrl Unit	4	0	Chan 0	Chan 1	Chan 2	Chan 3	Chan 4	Chan 5	Chan 6	Chan7
	4	1								
MPX Channel	4	2								
2870 MPX Channel	4	2		RPQ - MPX S/C 000=No additional 001=1 " 010=2 " 011=3 " 100=4 "				3 bit meaning 000=No selector chns 001= 1 selector chn 010= 2 " " 011= 3 " " 100= 4 " "		
Hi-speed MPX Channel	4	3						3 bit meaning 000=No subchannels 001= 1 subchannel 010= 2 subchannels 011= 3 subchannels 100= 4 subchannels		
Selector Channel	4	4								
Selector Channel RPQ	4	5								Apollo sim.
Storage Channel	4	6								
	4	7								
	4	8								
	4	9								
	4	A								
	4	B								
	4	C								
Chan to Chan	4	D								
Direct Control Chan	4	E								
Direct Data Feature	4	F								

	Unit Type		Options (Hex digit 1)				Options (Hex digit 2)			
bits	0-3	4-7	0	1	2	3	4	5	6	7
2400(or 2415) tape (speeds are noted instead of models for IRG tests)	5	0			2 bit meaning 00= 18.75 ips 01= 37.5 10= 75 11=112.5		2 bit meaning 00=NRZI only 01=Expansion 10=P.E. &NRZI 11=P.E. only			Data converter
Simultaneous 2404/2408 Rd/Wr Tape Control	5	1	Same options as for 2400, unit type 50							
TIC (Tape Interconnecting Unit)	5	2								
Hypertape	5	3	ASCII							
Tape/Hypertape Switch - 2816	5	4	Switch Number 00=1st sw. 01=2nd sw. 10=3rd sw. 11=4th sw.		CPU No. 00=CPU 1 01=CPU 2 10=CPU 3 11=CPU 4		Number of drives attached 0000 = 1 drive to 1111 = 16 drives			
2803 Dual Interface	5	5								
	5	6								
	5	7								
	5	8								
	5	9								
	5	A								
	5	B								
	5	C								
	5	D								
	5	E								
	5	F								

	Unit Type		Options (Hex digit 1)				Options (Hex digit 2)			
bits	0-3	4-7	0	1	2	3	4	5	6	7
2311 Serial File	6	0	ASCII	Record O'flow	1302 Stge	File Scan	2 Chan Switch			
2841	6	1								
2302 Disk	6	2	ASCII	Record O'flow	Addtl 1302 Stg	File Scan	2 Chan Switch			
	6	3								
2321 File	6	4								
	6	5								
2303 Drum	6	6	ASCII	Record O'flow		File Scan	2 Chan Switch			
	6	7								
2841 Two Chan Switch see program write-up	6	8	<u>Switch Number</u> 00=1st sw. 01=2nd sw. 10=3rd sw. 11=4th sw.		<u>CPU Number</u> 00=CPU 1 01=CPU 2					
2301 Drum	6	9	ASCII							
2314 or 2844 File	6	A								
2822 Two Chan Switch See program write-up	6	B								
2318 File	6	C								
	6	D								
2310 File	6	E								
2973-2 File Switch See program write-up	6	F								

	Unit Type		Options (Hex digit 1)				Options (Hex digit 2)			
bits	0-3	4-7	0	1	2	3	4	5	6	7
2260/2848 Display See program write-up and Fig. 13.	7	0	1053 Feat.	1053 Printr	2 bit meaning 00 = 960 size 01 = 480 size 10 = 240 size		2 bit meaning 00=No keybrd 01=Numeric 10=Alphabetic 11=Data entry		Line Addr.	Non-Destr. Cursor
XY Plotter	7	1								
Administrative Processor Console	7	2								
Cypress	7	3								
1015	7	4	ASCII	Plotting Character						
2250 I Unbuffered	7	5	ASCII	Character Gen.	Vector Graph		Light Pen	A N Key Board	P F Key Board	
2250 I Buffered	7	6	ASCII	Character Gen.	Vector Graph	Xpand Size Buffr	Light Pen	A N Key Board	P F Key Board	
2840/2250 See Fig. 13. See program write-up	7	7	ASCII		Vector Graph	Xpand Size Buffer 2840I -16K 2840II -32K	Light Pen	A N Key Board	P F Key Board	
	7	8								
2840/2280-2 Alpine	7	9	ASCII	2280		Xpand Size Buffer 2840I -16K 2840II -32K				
	7	A								
	7	B								
	7	C								
	7	D								
	7	E								
2814 Two Chan Switch for 2250-1/2840/2848	7	F	2250-1 Unbfrd	2250-2 Unbfrd ANK PFK LP	2848/2260 No key board	2848/2260 Keybrd	2840/2280			

	Unit Type		Options (Hex digit 1)				Options (Hex digit 2)			
bits	0-3	4-7	0	1	2	3	4	5	6	7
2821 Control	8	0								
2540 Reader	8	1		Card Image		51 Col Feat.	1400 Feat.			2 Chan Sw on 2821
2540 Punch	8	2	ASCII	Card Image	Punch feed read	51 Col Feat.	1400 Feat.			2 Chan Sw on 2821
1403 Printer	8	3	ASCII	Sel. Tape Lister	60 Char. Set	Univ. Char. Set	High speed Printr	120 Pos.		2 Chan Sw on 2821
2203	8	4								
1404	8	5		ASCII	Rd-Cmp					
2821 Two Chan Switch See program write-up	8	6								
1442	8	7	ASCII	Card Image						N2 Punch
1443 Printer	8	8	ASCII	144 Pos.						
1445	8	9								
2211	8	A		Sel Tape List				144 Pcs.		2 Chan Sw
	8	B								
2501	8	C	ASCII	Card Image						
2520	8	D	ASCII	Card Image						2520 B2&B3 Punch
2560 MFCM	8	E								
	8	F								

	Unit Type		Options (Hex digit 1)				Options (Hex digit 2)			
bits	0-3	4-7	0	1	2	3	4	5	6	7
1052	9	0	Tab Bksp Line Feed	Ribbon Shift						
Reserved for DEU	9	1								
Paper Tape Reader	9	2	<u>4 bit meaning</u> 0000=2671 Basic Unit 0001=Olivetti or 6 track Japanese tape 0010=2671/2822 & 2 chan sw. 0011=Facit paper tape reader							
2672 Punch	9	3								
	9	4								
	9	5								
	9	6								
	9	7								
1827 ADC	9	8								
1827 MPX	9	9								
1827 Comparator	9	A								
1827 Digital input	9	B								
1827 Digital Analog output	9	C								
	9	D								
	9	E								
	9	F								

	Unit Type		Options (Hex digit 1)				Options (Hex digit 2)			
bits	0-3	4-7	0	1	2	3	4	5	6	7
1419 Single Address	A	0	Prog Lights 4-7	Mod 32	Batch Number	Dash Xmissn	Prog Lights A-3	Binary value equals external sig. line # 0=no external signal		
1419 Dual Address See program write-up	A	1	Same options as 1419, unit type A0							
1418	A	2	ASCII	3 Pocket	Sta 1 Instld	Mrk Rd -Sta 1		Binary value equals external sig. line # 0=no external signal		
	A	3								
1412	A	4	ASCII			Dash Xmissn		Binary value equals external sig. line # 0=no external signal		
	A	5								
1428	A	6	ASCII	3 Pocket	Sta 1 Instld	Mrk -Sta 1		Binary value equals external sig. line # 0=no external signal		
	A	7								
1285	A	8								NCR
1231	A	9								
1287	A	A								
	A	B								
	A	C								
	A	D								
	A	E								
	A	F								

	Unit Type		Options (Hex digit 1)				Options (Hex digit 2)			
bits	0-3	4-7	0	1	2	3	4	5	6	7
2701 SSYA	B	0								
2701-1050 1060 1070 Adapter (SSA1)	B	1	ASCII	134.49 BPS Line Speed	600 BPS Line Speed	150 BPS Line Speed	Auto- Dial	1050 TTY Line Adaptr	0=Line Leased 1=Sw. Netwrk	
2701- TTY 28 Adapter (SSA2)	B	2	ASCII	45.5 BPS Line Speed	56.89 BPS Line Speed	74.2 BPS Line Speed				
2701 33-35	B	3								
2701 1030 adapter	B	4								
2701 2 Processor Sw	B	5								
2701-WTC TTY Adapter (SSA6)	B	6	ASCII	50 BPS Line Speed	75 BPS Line Speed					
2701 Contact Sense Adapter	B	7	ASCII	Exten- sion Fea- ture 1	Exten- sion Fea- ture 2	Time Out Delay				
2701 Control Adapter	B	8								
2701 Parallel Data Adapter	B	9	ASCII	Time Out Delay	Number of expansion Features					
2701 Synchronous Data Adapter	B	A		Dual Comm. Inter- face	Early End	Line Speed 00=Data Set 01=Clock X 10=Clock Y 11=Clock Z		Line Type 00=2 Wire 1/2 Duplex 01=4 Wire 1/2 Duplex 10=Full Duplx		Auto Dial
	B	B								
2701 IBM Terminal Adapter	B	C								
2701 Binary Synchr. Adapter	B	D								
	B	E								
2701 Synchr Adapter with 3945-8	B	F								

	Unit Type		Options (Hex digit 1)				Options (Hex digit 2)			
bits	0-3	4-7	0	1	2	3	4	5	6	7
1009	C	0								
1013	C	1								
7701	C	2								
7702	C	3								
2020 Comm. Adapter	C	4								
2702 Transparent Teletype	C	5								
2701 IPCA See program writeup	C	6								
3945-8	C	7								
2702 or 2702 Quantas See Fig. 13 and program writeup	C	8			0000=ignored except in EC81 0001=IBM Type 1 0010=IBM Type 1 Autocall 0011=IBM Type 2 0100=TTY Type 1 0101=TTY Type 2 0110=TTY Type 1 Autocall 0111=WTC TTY 1001=IBM Type 1&Break feat.				00= SAD 0 01= SAD 1 10= SAD 2 11= SAD 3	
7770 2 Processor Sw	C	9								
7772	C	A	No. of lines 00=2 01=4 10=6 11=8		0=Wrld Trde 1=Domestic	0=DT 1=3/14 code				
2703	C	B								
	C	C								
	C	D								
	C	E								
	C	F								

	Unit Type		Options (Hex digit 1)				Options (Hex digit 2)			
bits	0-3	4-7	0	1	2	3	4	5	6	7
	D	0								
1030/1050/1060/ 2740/2741	D	1					0000=1030 0001=1050 0010=1060 0100=2740 0101=2741			
	D	2								
10MD	D	3								
1070	D	4								
Model 28 TTY	D	5								
Model 33 TTY	D	6								
Model 35 TTY	D	7								
IBM Teletype	D	8								
WTC Telegraph	D	9								
	D	A								
2260/2848 Remote Start/Stop	D	B	DAT records required. See program write-up.							
2250 Remote Terminal	D	C								
	D	D								
	D	E								
	D	F								

	Unit Type		Options (Hex digit 1)				Options (Hex digit 2)			
bits	0-3	4-7	0	1	2	3	4	5	6	7
	E	0								
1012	E	1								
	E	2								
2702 8A1 Teletype	E	3								
2701 Autodyne ASCII	E	4								
2701-1006 Full Duplx	E	5	Rcvr Adptr	Xmtr Adptr	6 bit meaning BCD code (BA8421) Interchange Addr.(Poll) of Receive Adptr						
2702-Boeing Data Collection	E	6								
2906	E	7								
2905	E	8								
	E	9								
	E	A								
	E	B								
	E	C								
	E	D								
	E	E								
	E	F								

Note: Check program write-ups of RPQ programs for more option bit details.

UNIT CODES FOR 8 BYTE ENTRY

The first 4 bytes of these unit entries are described in the previous 4 byte unit definition e charts. Only the second 4 bytes are shown here. Always refer to program descriptions for fur details, since these charts do not necessarily show latest revisions.

Unit Type Byte 1		Byte 2	Bytes 3&4	Byte 5	Byte 6 Digit 1	Byte 6 Digit 2	Byte 7 Digit 1	Byte 7 Digit 2	Byte 8 Digit 1	Byte 8 Digi
		Op-tion fig. 12	Chan and Unit Addr	DM Byte	Hexadecimal representation by bits: 8 4 2 1	8 4 2 1	8 4 2 1	8 4 2 1	8 4 2 1	8 4 2
2260/ 2848	70			Must be 00	Punch hex sum 0-F 8=Selector interface				Low order 2260/2848 address	
2250 /2840	77			Must be 00	Punch hex 8=Graphic Design 4=32K buffr 2=2 usec buffer		Punch hex 8=GM RPQ Pen sw open 2=Four level intensity	Punch hex 1=Blink mode (RPQ)		
2821 Two Chan Sw	86			Must be 00	Punch hex sum 0-F 8=Printer 1 4=Reader 2=Punch 1=Printer2	Punch hex sum 0-F 8=Printer 3	Punch hex 8=MPX Chan 4=Sel 1 2=Sel 2			
2702 (WTC TTY only)	C8			Must be 00	Figs-X character		Figs-Y character			
					This entry is only required for those systems on which SIP is used					

Figure 13: Unit and Feature Codes for 8 Byte Entry

Diagnostic Program Report

Please indicate any problems, errors, omissions, or desired improvements in the entire diagnostic program package, you received.

Describe the problem. Indicate:

Program name or document -
EC level -
DM used -

System configuration(that program was run on) which caused problem.

Did you find a solution? If you did, what is your suggested solution?

How can the diagnostic program package be improved?

Submitted by: (please use your mailing address)

Name __________

Address __________

Investigated by: (for Diag. Eng. use)

Name ____________
Dept. ___

Action taken:

FOLD, STAPLE, AND MAIL

fold fold

FIRST CLASS
PERMIT NO. 81
POUGHKEEPSIE, N.Y.

BUSINESS REPLY MAIL
POUGHKEEPSIE, N.Y. 12602

POSTAGE WILL BE PAID BY

IBM CORPORATION
P.O. BOX 390
POUGHKEEPSIE, N.Y. 12602

ATTN: DIAGNOSTIC ENGINEERING
DEPT. B75

---

fold fold

SR23-3112-0
Courses:
11271
41242
52150
52152

International Business Machines Corporation
Field Engineering Division
360 Hamilton Avenue, White Plains, N.Y. 10601

IBM World Trade Americas/Far East Corporation
Town of Mount Pleasant, Route 9, North Tarrytown, N.Y., U.S.A. 10591

IBM World Trade Europe/Middle East/Africa Corporation
360 Hamilton Avenue, White Plains, N.Y., U.S.A. 10601

System/360 Diagnostic Program General Reference Manual Supplementary Course Material Printed in U.S.A. SR23-3112-0