DPS 6 GCOS 6
DATA BASE AUGMENTED
REAL-TIME TRACING SYSTEM
USER'S GUIDE

SUBJECT

Procedures Used in Operating the Data Base Augmented Real-Time Tracing System

SPECIAL INSTRUCTIONS

This manual supersedes CZ74-00, dated April 1983. This manual has been extensively revised and reorganized; therefore, change bars and asterisks are not used.

SOFTWARE SUPPORTED

This manual supports Release 4.0 of the Data Base Augmented Real-Time Tracing System User's Guide, running under Release 4.0 of the MOD 400 Executive.

ORDER NUMBER

CZ74-01

March 1986

Honeywell

Base Publication Number	Manual Title
HC01	MOD 400 Application Development Overview
HC12	Disk-Based Data Entry Facility-II User's Guide
HC13	Disk-Based Data Entry Facility-II Operator's Quick Reference Guide

The following manuals describe the MOD 400 distributed processing software components:

Base Publication Number	Manual Title
CB35	DPS 6/DPS 7 PVE File Transfer Facility User's Guide
CF11	DPS 6/DPS 7 PVE Remote Batch Facility User's Guide
CG90	Interactive Entry Facility-II User's Guide
CZ59	Level 6 to Level 6 File Transmission Facility User's Guide
CZ60	Level 6 to Level 66 File Transmission Facility User's Guide
CZ61	Level 6 to Level 62 File Transmission Facility User's Guide
CZ62	BSC Transport Facility User's Guide
CZ63	2780/3780 Workstation Facility User's Guide
CZ64	HASP Workstation Facility User's Guide
CZ65	Programmable Facility/3271 User's Guide
CZ66	Remote Batch Facility/66 User's Guide
GG19	Disk-Based VIP7305 Emulator Facility User's Guide
GG20	Asynchronous Communications Facility User's Guide
GT18	Disk-Based VIP7705 Emulator Facility User's Guide
GT19	Disk-Based VIP7814 Emulator Facility User's Guide

The following manuals describe the ORACLE data base management facility:

Base Publication Number	Manual Title
GS61	GCOS 6 MOD 400 ORACLE Installation Guide
GS62	GCOS 6 MOD 400 ORACLE Data Base Administrator's Guide
GS63	GCOS 6 MOD 400 ORACLE Interactive Application Facility (IAF) Terminal Operator's Guide
GS64	GCOS 6 MOD 400 ORACLE Interactive Application Facility (IAF) Terminal Operator's Reference Manual
GS65	GCOS 6 MOD 400 ORACLE Interactive Application Facility (IAF) Designer's Guide
GS66	GCOS 6 MOD 400 ORACLE Interactive Application Facility (IAF) Designer's Reference Manual
GS67	GCOS 6 MOD 400 ORACLE HLI Precompiler Interface
GS68	GCOS 6 MOD 400 ORACLE Host Language Call Interface Manual
GS69	GCOS 6 MOD 400 ORACLE RPF Report Text Formatter User's Guide
GS70	GCOS 6 MOD 400 ORACLE RPT Report Generator User's Guide
GS71	GCOS 6 MOD 400 ORACLE SQL/UFI Reference Manual
GS72	GCOS 6 MOD 400 ORACLE Terminal User's Guide
GS73	GCOS 6 MOD 400 ORACLE Utilities Manual
GS74	GCOS 6 MOD 400 ORACLE Error Messages and Codes

In addition, the following publications provide supplementary information:

Base Publication Number	Manual Title
AS22	Level 6 Models 6/34, 6/36, and 6/43 Minicomputer Handbook
AT97	Level 6 Communications Handbook
CC71	Level 6 Minicomputer Systems Handbook
CD18	Level 6 MOD 400/600 Online Test and Verification Operator's Guide
FQ41	Writeable Control Store User's Guide

These five manuals are not covered by the Guide to Software Documentation. See your Honeywell representative for information concerning the versions of the manuals relevant to your installation.

Users should be aware that a software release bulletin accompanies each software product ordered from Honeywell. Users should consult the software release bulletin before using the software. Users should contact their Honeywell representative if a copy of the software release bulletin is not available.

CONTENTS

CONTENTS

CONTENTS

ILLUSTRATIONS

Section 1
OVERVIEW

This section provides a general overview of the Data Base Augmented Real-Time Trace System (DARTS).

DARTS is an online software trace facility available with MOD 400 Executive and networking software. It is a field support and maintenance facility.

DARTS FACILITY

DARTS monitors the execution of a set of software modules operating under the MOD 400 system. The user defines and operates Trace Information Points (TIPs) at specific sites within the modules in question. The user also builds a Trace Information Capture Specification (TICS) for each TIP so that specific information can be collected when the TIP executes.

See Section 2 for a description of how DARTS monitors the flow of execution using TIPs and TICS.

DARTS CONFIGURATION AND INITIALIZATION

A Configuration Load Manager (CLM) LDBU directive is a prerequisite for using DARTS. Configuring MOD 400 activates the directive. When brought online DARTS requires a data base file; the data base file is processed during initialization.

Section 3 and Appendix B provide detailed instructions for bringing DARTS online.

COMMAND LANGUAGE

The DARTS facility accepts commands from three sources:

1. The initialization data base file
2. The operator console (interactive)
3. The XEC command file (noninteractive).

The format of a command is the same regardless of the source. The commands can be used to define and control the TICS, the DARTS buffer pool, and the logging and formatting operations.

Sections 4 and 5 describe the command language in detail.

Section 2
DARTS MONITOR

TRACE INFORMATION POINTS

A Trace Information Point (TIP) represents a user interface to the DARTS Data Collector. A TIP must be predefined and assembled in line with the user software. Each TIP must be identified by a unique hexadecimal four-digit (16-bit) number. The leading two digits (Module Code) cannot be 00. (Refer to Section 4 for allocated Module Codes.) A defined TIP without an associated Trace Information Capture Specification (TICS) is an inactive TIP.

A set of internal buffers holds the data collected for each TIP in chronological order. The time-stamped data buffers are logged onto a disk file.

Commands can be issued to the DARTS monitor from the operator's console. The user can issue commands to define new TICS, manipulate the execution of defined TICS, control the logging of the raw data (time stamped), and format the logged data.

Format either online or offline; that is, either while the data collection takes place or after the data collection session has ended.

Predefined TIPs and TICS

A set of TIPs is predefined in some software modules. These TIPs are assembled inline, using one of the two interfaces described later in this section. The associated TICS are specified by a set of standard DARTS user commands contained in data base files (see Appendix B) usually residing under the >>SID directory.

The information collected at the predefined TIPs allows software support personnel to view the operation of the software modules in question. By selective activation of these TIPs, information can be filtered at the time of collection. By logging a minimum amount of raw data, the user can follow the flow of execution (and data) in chronological order across layers of software.

DARTS FACILITY

The DARTS monitor consists of the following components:

- Data Collector
- Command Manager
- Command Processors
- Data Buffer Manager
- Data Buffer Logger
- Log File Formatter.

When invoked with an Operator Control Language (OCL) command, DARTS operates in the system pool ($$) in both distributed and nondistributed environments. DARTS can trace and format collected data only when invoked using an OCL command within DARTS. Optionally, DARTS can be invoked with an ECL command from a user task group acting strictly as the formatter of the specified log file.

Data Collector

When a TIP executes, the data collector is responsible for collecting data into one or more of the internal buffers maintained by DARTS. The collector operates in a distributed environment and executes at the level of the caller. The data collector uses the 16-bit TIP id to locate the associated TICS structure. From information in the TICS, the data collector determines whether or not data is to be collected, and the extent of the data to be collected.

When a buffer is filled the data collector logs it. Data collection at a TIP takes place only when the logging operation is active.

User Interfaces to the Data Collector

Two types of user interface to the data collector are supported at a TIP:

1. A general (trap) interface, available to all users, through a Trace instruction at a TIP. The Trace instruction (Z'1234') generates an illegal instruction trap (05) which is intercepted by a special trap handler to perform the trace function.

2. A special (link-and-jump) interface, available only to system level modules for faster execution at a TIP. Users of this interface must supply a 64-word stack for the DARTS collector.

Software operating in Ring 0 can use either interface, while software executing in Rings 1/2/3 is restricted to the general interface.

Command Manager

The command manager component of DARTS parses the interactive operator command entered from the system console and schedules the appropriate command processor.

Interactive input is processed at the level of the DARTS (OCL) task. While individual DARTS commands can be issued by the operator interactively, putting individual DARTS commands into a data base file and issuing DARTS commmands via the XEC command (refer to Section 4) is recommended.

Command arguments are parsed into a parameter block and passed to the specific command processor overlay, which is loaded, executed, and then unloaded.

Command Processors

Specific command processors are available as floatable overlays of the DARTS bound unit, to be loaded as necessary for one-time execution. A link-and-jump interface exists between the command manager and a command processor.

Data Buffer Manager

The data buffer manager maintains the chain of internal buffers and acquires additional buffers as the need arises. It acts in concert with the data buffer logger as a nondistributed task, and is created as part of DARTS initialization. This task ends when DARTS terminates.

Data Buffer Logger

The data buffer logger processes all buffers scheduled by the Data Collector. It writes the scheduled buffers to the log file in chronological order. It operates with the data buffer manager in a nondistributed environment. The data collector triggers the data buffer logger.

The data buffer logger operates on a file created by a LOG command.

Log File Formatter

An interactive DARTS command schedules the log file formatter. The log file formatter formats the collected data and puts the data in designated log files. Command arguments allow for the processing of data collected in a specified time frame.

Section 3
CONFIGURATION, INITIALIZATION, AND TERMINATION

CONFIGURATION

The DARTS facility can be used on a MOD 400 4.0 system only after a configuration component has been loaded by the Configuration Load Manager (CLM). This is done implicitly, provided the bound unit, ZXDJ, is located under the >>SID directory.

The configuration component contains the preprocessor of the data collector for the general and special interfaces of the DARTS Trace instructions. The preprocessor replaces minimal processor routines present in the Executive.

The presence of the configuration component in a system is reflected in the System Control Block (SCB). At offset S_DINF, DARTS sets bit 0 indicating that it has been configured (bit 0 represents a mask of X'8000'). The template macro for the SCB (Z3SCB), contained in the OS_LIB macro library, defines the offset S_DINF.

INITIALIZATION

Initialize DARTS by an OCL command specifying an initialization data base file. The DARTS data collection component, as well as the buffer manager component and buffer logger component, loads into the system as part of initialization. Upon completion of initialization, bit 1 of offset S_DINF is set in SCB (mask of X'4000').

Initialize DARTS using:

```
$S DARTS [-D  path] [-E] [-A argl... arg9]
```

where:

-D path

Specifies the absolute pathname of the selected initialization data base file. The default is >>SID>BULLSEYE.

-E

Causes each command input to be echoed at the operator's console. Use of -E is optional.

-A argl... arg9

Specifies between 1 and 9 optional arguments passed to the command processor executing DARTS commands in the data base file. (For details refer to "Execute Command File (EXC)" in Section 4.)

Initialization Process

Four major steps occur automatically in the initialization process:

1. The establishment of the data collector.
2. The creation of the Data Buffer Manager/Logger task. It is activated when the BFR and LOG commands are processed.
3. A first pass at the commands in the data base file to generate the TICS structures for the predefined TIPs. Some or all of the TIPs can be activated.
4. A second pass at the contents of the data base file to record each command onto the online log file (assuming that the log file has already been established by a LOG command).

The DARTS collector is fully operational after completing these steps.

An nonrecoverable error in the execution of a command in this data base file automatically aborts the DARTS startup, deleting all constructed structures. Once the error has been corrected, DARTS can be reinitialized with another OCL command without adverse effect on the operation of other elements in the MOD 400 system.

Data Base File

The data base file is a sequential file set up by the user prior to the initialization of DARTS. It contains user commands that are processed sequentially during initialization to generate the execution environment for DARTS. User commands can configure the set of internal buffers, establish the log file, and specify TICS structures for all TIPs, or a relevant subset of TIPs. (Refer to Appendix B for recommendations and an example.)

Use the MOD 400 Line Editor (ED), or Screen Editor (Scorpeo), to create and modify the data base file. (Refer to the GCOS 6 MOD 400 Application Developer's Guide (Order No. CZ15) for a detailed description of the Editors.) Each command in this file must be specified as a single record of up to 256 bytes. No DARTS command specifically restricted to use from the operator's console can be present in the data base file. (For an example refer to "Execute Command File (XEC) in Section 4.)

TERMINATION

Terminate the DARTS facility using the OCL command:

```
$S DARTS -Q
```

The command gracefully shuts down all on-going collection and logging activities, closes the log-file, and deletes all internal structures and buffers. DARTS can be brought online again with another OCL initialization command.

Section 4
COMMAND LANGUAGE

USER INTERFACES

The user has two interfaces to DARTS, for two entirely different purposes:

1. USER TIP INTERFACE: A Trace Point interface to the data collector for use at each Trace Information Point (TIP).

2. USER COMMAND INTERFACE: A command (OCL) interface to the command manager that defines the Trace Information Capture Specification (TICS) for a TIP and controls the operation of the DARTS trace facility.

USER TIP INTERFACE

The user can choose between two distinct types of TIP interface when setting up a TIP:

1. A General TIP interface available to all users

2. A Special TIP interface available only to users executing at ring 0 on the DPS 6 system. Typically, these are system tasks executing in the system pool ($$).

There are two macros available in the MOD 400 system EXEC_LIB macro library for defining TIPs within a user's application:

1. $tipsp

   Specifies an environment in which a $tip macro generates the instruction sequence of a special interface TIP. The $tipsp macro itself generates no instructions.

2. $tip

   Generates a Trace Point instruction sequence of either a special or a general interface TIP, depending on whether or not the $tipsp macro precede it.

General TIP Interface

This interface is built around the use of a trace instruction to request the services of the DARTS data collector. The one-word trace instruction (Z'1234') causes an illegal instruction trap that is intercepted by a special trap handler and in turn processed by the data collector.

Provided that the $tipsp macro or an equivalent macro has not preceded $tip, the general (trap) interface can be arrived at by using the macro:

```
$tip  id
```

where:

id

Represents the 16-bit TIP id specified as a four-character hexadecimal string. Refer to "TIP Identification Code" in this section.

The macro generates the one-word trace instruction, followed by a one-word (16-bit) TIP id field. For example:

```
DC  Z'1234'
DC  Z'hhhh'
```

Note that hhhh signifies a four-character hexadecimal string.

Special TIP Interface

This interface is primarily reserved for system tasks. The user of this interface is responsible for supplying the collector with a stack of at least 128 bytes (pointed to by $B7) using register $B5 as the linkage register. The System Control Block (SCB) links the stack to the data collector. Offset S_DART of the SCB contains the entry point address of the preamble to the DARTS data collector.

When the $tipsp macro or an equivalent macro precedes $tip, the special interface can be arrived at by using:

```
$tip  id[,offset]
```

where:

id

Is the 16-bit TIP id, specified as a four-character hexadecimal string. The TIP id field is generated following the link-and-jump instruction.

offset

An optional specification used only when register $B6 points to a structure initialized with the address of the data collector at the specified offset. The generated instruction sequence is:

```
LNJ   $B5,*$B6.offset
DC    Z'hhhh'
```

Note that hhhh signifies a four-character hexadecimal string.

When offset is omitted, the generated instructions make exclusive use of register $B5 to link to the data collector through the SCB. The instruction sequence is:

```
LDB   $B5,<ZHCOMM+X'18'
LNJ   $B5,*$B5.S_DART
DC    Z'hhhh'
```

The offset S_DART, defined in the SCB template macro (Z3SCB), appears in the OS_LIB macro library.

Predefined TIPs

Users of standard software modules are encouraged to use predefined TIPs, using either the general or the special TIP interface. Tracing the execution of the module can be made standard with a static arrangement of TIPs of known identity.

The information collected at these predefined TIPs can be specified by DARTS commands. This permits flexibility in the selection of the information captured at each TIP.

TIP Identification Code

A DARTS TIP, whether it uses the general or the special interface, must have a 4-digit hexadecimal (16-bit) identification code unique in the domain of the DARTS facility; i.e., the MOD 400 system.

The TIP id digits are interpreted as:

mmss

where:

mm

Represents the two-digit module code.

ss

Represents the two-digit site code.

Specific module codes are assigned to standard software modules incorporating predefined TIPs, with multiple TIPs in a module being identified by unique site codes.

Module Codes Reserved for User Applications

A module code of 00 is reserved for use by DARTS. Module codes E0 through FF are reserved for the use of customer application software modules.

USER COMMAND INTERFACE

The user command interface defines the format of the commands directed at DARTS from either the operator's console or a command file (such as the data base file). The command format is as follows:

```
$S DARTS xxx argument_list
```

where:

xxx

Represents the three-character mnemonic of a DARTS command.

argument_list

Specifies an argument list consisting of one or more positional and keyword arguments. Keyword arguments use a single-character mnemonic prefixed with a hyphen. DARTS commands in the data base file use the same format as commands from the operator's console. Certain commands can only be issued from the operator's console.

The maximum length of a command is 256 bytes. When issued from the operator's console, a command spans input lines by using a continuation character (&).

HELP Messages

A HELP message appears when a command is entered without an argument. Commands requiring no argument are exceptions to this rule: DBF, SLG, RLG, SFM, RFM and TFM.

Command Execution

There are three sources of commands to the DARTS facility:

1. The initialization data base
2. A command issued from the operator's console
3. A command contained in a file being processed by the XEC command.

The entire DARTS command set is available to the user from the operator's console. Some commands cannot be issued from the data base (during initialization), or from a command file processed as a result of the XEC command. These interactive-only commands appear in the list that follows.

With the exception of a formatter request, every command executes synchronously. Each command processor is part of an overlay of the DARTS bound unit. Each loaded overlay, executes and unloads in turn.

User Commands

A list of user commands follows. Some commands can only be used in the interactive mode, as noted.

TICS DEFINITION COMMANDS

The following is a list of TICS definition commands:

ABT	Abort TICS Definition	Aborts the definition in progress of a TICS (interactive only)
DAT	Data Array Definition	Defines data array to be collected
DEF	Define TICS	Begins the definition of a TICS for a TIP
DTC	Display TICS Attributes	Displays the attributes of a TICS for a TIP (interactive only)
END	End TICS Definition	Ends the definition of a TICS
REG	Register Set Definition	Register set to be collected is defined
REM	Remove TICS Definition	Removes the definition of a TICS for a TIP
TOE	TICS Table of Entries	Defines Table Of Entries for TICS

TICS MANIPULATION COMMANDS

The following is a list of TICS manipulation commands:

ACT	Activate TICS	Activates a defined TICS
DEA	Deactivate TICS	Deactivates a defined TICS

RCX	Reset Condition for Execution	Resets the Condition for Execution of a TIP
SCX	Set Condition for Execution	Sets the Condition for Execution of a TIP

BUFFER POOL COMMANDS

The following is a list of buffer pool commands:

BFR	Buffer Pool Definition	Defines buffer pool attributes
DBF	Display Buffer Pool	Displays buffer pool status (interactive only)

LOGGER CONTROL COMMANDS

The following is a list of logger control commands:

LOG	Log File Definition	Defines log file to be used
RLG	Resume Logging	Resumes logging operation (interactive only)
SLG	Suspend Logging	Suspends logging operation (interactive only)

FORMATTER COMMANDS

The following is a list of formatter commands:

FMT	Format Logged Data	Schedules formatting of logged data (interactive only)
RFM	Resume Formatting	Resumes ongoing formatting (interactive only)
SFM	Suspend Formatting	Suspends ongoing formatting (interactive only)
TFM	Terminate Formatting	Terminates ongoing formatting (interactive only)

GENERAL COMMANDS

The following is a list of general commands:

*	Comment line, for use in Command files	
XEC	Execute Command File	Executes commands in a Command file (interactive only)

The remainder of this section provides a detailed description of each command listed alphabetically by title.

ABORT TICS DEFINITION (ABT)

Aborts the definition in progress of a TICS.

FORMAT:

ABT id

ARGUMENTS:

id

A four-digit hexadecimal string TIP identifier defining the TICS being aborted. The left pair of digits represents a module code and the right pair represents a site code within the module. 00 cannot be used as a module code.

NOTE

Use ABT only on a TICS definition in progress. For a defined TICS refer to the REM command.

DESCRIPTION:

This command can only be issued in the interactive mode.

Example:

ABT E001

Aborts the definition in progress of the TICS for TIP id E001.

ACT

ACTIVATE TICS (ACT)

Activates the specified list of defined but inactive TICS.

FORMAT:

```
ACT  list
```

ARGUMENT:

list

A list of one to eight TIP id elements, separated by spaces. Each TIP id element can be either a single id, including wildcards, or an id range specified as low/high. A TIP id is a four-digit hexadecimal string (mmss) specifying a TICS. (Refer to "TIP Identification Code" in this section.)

A wildcard id of an asterisk (*) can activate all currently defined TICS. TICS of a specific module code can be activated using mm* as the id, where mm is the two-digit hexadecimal module code, or using mms* as the id, which implies activation of ids in the range mms0 and mmsF.

Do not use a module code (mm) of 00.

DESCRIPTION:

The specified TICS is activated.

Example 1:

```
ACT  *
```

Activates all currently defined TICS.

Example 2:

```
ACT EE* EF05
```

Activates all currently defined TICS for TIPs with module code EE and activates the TICS for the TIP EF05.

Example 3:

```
ACT EE07/EE15
```

Activates all currently defined TICS with TIP ids of EE07 through EE15, inclusive.

BUFFER POOL DEFINITION (BFR)

Defines the attributes of the buffer pool used by DARTS.

FORMAT:

```
BFR  size number low_limit
```

ARGUMENTS:

size

The specified buffer size. It is rounded up to the next multiple of 256 bytes with a maximum size of 4096 bytes.

number

The initial number of buffers allocated.

low_limit

The threshold at which the buffer manager acquires additional buffers to satisfy peak activity demand. The low_limit must be less than the number of initially allocated buffers. The total number of allocated buffers cannot exceed 2 * number.

DESCRIPTION:

This command must be issued before the LOG command since some characteristics of the log file are determined by the buffer size in use.

Example:

```
BFR  1000 7 4
```

Specifies internal buffers of 1024 bytes (next multiple of 256) with an initial count of seven buffers and a low threshold of four buffers. (The total number of acquired buffers cannot exceed 2 * 7 = 14 at any time.)

NOTE

Adjust the size and number of allocated buffers according to the rate of trace information being collected. Although the collected data can span several buffer blocks, buffers should be sized such that one block is large enough to accommodate the data from the TIP generating the most output.

DAT

DATA ARRAY DEFINITION (DAT)

Defines one data array entry (of up to four) in the TICS being defined.

FORMAT:

```
DAT address range [-L label] [-M limit] [-O]
```

ARGUMENTS:

address

The data array's address is computed from the prevailing register context ($Bx and $Ry) at the TIP. The specified address field can only be in one of the four Assembly-language-type formats listed below:

Base Register addressing (x=1-7)

```
 $Bx.OFFSET
*$Bx.OFFSET
 $Bx.$Ry     (y=1-7)
*$Bx.$Ry     (y=1-7)
```

The asterisk (*) represents indirect addressing. Indexed addressing allows for the use of all seven $R registers.

OFFSET is a 16-bit signed value that must be specified as a decimal literal. It cannot be an expression.

range

Can be expressed as a decimal literal ranging from 1 to 2000 bytes. A value in excess of 255 bytes is rounded up to the next multiple of 8 bytes.

Alternatively, range can be specified as a $Rx register (x=1-7) for dynamic ranging based on the contents of the register. Range can also be specified as the contents of a location addressed by the syllable $Bx.OFFSET. (Note that indirect addressing is not allowed in range addressing.)

A limit of 2000 bytes of data collected is imposed with dynamic ranging. A lower limit can be specified (see example 1).

-L label

This argument is optional and can be used to specify a one- to eight-character string to be used by the formatter when outputting individual data arrays. It is recommended that it be specified. If omitted, eight blanks are used as the label.

-M limit

This argument is optional. It can be expressed as a decimal literal. The argument enforces a limit on data collected with dynamic ranging. If greater than 255 bytes, the limit is rounded down to a multiple of 8 bytes.

-O

This optional argument indicates that data collection begins with the right byte of the first location in the array being addressed (odd-byte addressing).

DESCRIPTION:

A maximum of four data arrays can be specified for each TICS. A maximum of 8000 bytes of data can be collected.

Example 1:

```
DAT  *$B4.4 $R2 -L BUFFER -M 200
```

Specifies a data array, labeled BUFFER, that can be indirectly addressed by a pointer at offset 4 of a structure pointed to by $B4. The length of the array (in bytes) is specified by the contents of $R2 at the TIP. It is limited to 200 bytes in this example.

Example 2:

```
DAT  $B1 40 -L CODE
```

Specifies a data array of 40 bytes, labeled CODE, pointed to by $B1 when the TIP is executed.

DEA

DEACTIVATE TICS (DEA)

Alters the state of the specified list of TICS from active to inactive.

FORMAT:

```
DEA  list
```

ARGUMENT:

list

A list of one to eight TIP id elements separated by spaces. Each TIP id element can be either a single id, including wildcards, or an id range specified as low/high.

DESCRIPTION:

The specified list of active TICS is deactivated. This command is complementary to the ACT command. Refer to the ACT Command description for more details and examples.

Example:

```
DEA  E109
```

Deactivates TICS E109.

DEFINE TICS (DEF)

Starts the definition of a TICS.

FORMAT:

```
DEF  id
```

ARGUMENT:

id

A four-digit hexadecimal string that specifies a TIP id. The left pair of digits represents a module code and the right pair represents a site code in the module. 00 cannot be specified as a Module Code. (Refer to "TIP Identification Code" in this section.)

DESCRIPTION:

A TICS specification consists of the following commands:

DEF to start the specification of a TICS.
REG to define the register context (optional, single).
DAT to define any data array entries (optional, multiple).
END to close out the specification.

The DEF command begins and the END command finishes the definition of a TICS. The initial state of the newly defined TICS structure for a TIP is inactive.

The TICS can be activated by the ACT command and deactivated by the DEA command. While the definition is in progress, it can be aborted with the ABT command. Once defined, it can only be removed with the REM command.

Example:

```
DEF  EE0F
```

Starts the definition of the TICS for TIP id EE0F.

DISPLAY BUFFER POOL STATUS (DBF)

Displays the current status of the DARTS buffer pool.

FORMAT:

DBF

ARGUMENT:

None.

DESCRIPTION:

This command can only be issued in the interactive mode.

The status of the Buffer Pool is displayed at the operator's console in the form of relevant counts, such as: buffers in use, buffers available, minimum number of available buffers, etc.

DISPLAY TICS ATTRIBUTES (DTC)

Displays the current status of a defined TICS.

FORMAT:

DTC id

ARGUMENT:

id

A four-digit hexadecimal string that specifies a TIP id. The left pair of digits represents a module code and the right pair represents a site code in the module. 00 cannot be specified as a module code. (Refer to "TIP Identification Code" in this section.)

DESCRIPTION:

This command can only be issued in the interactive mode.

All the attributes of the TICS of the specified TIP id are displayed at the operator's console. The information includes the details of the REG and DAT commands, used to define the TICS, as well as the condition(s) for its execution, via the SCX command, if any.

END TICS DEFINITION (END)

Specifies the end of the definition of a TICS.

FORMAT:

```
END  id
```

ARGUMENT:

id

A four-digit hexadecimal string that defines the TIP in question. The left pair of digits represents a module code and the right pair represents a site code in the module. 00 cannot be specified as a module code. (Refer to "TIP Identification Code" in this section.)

DESCRIPTION:

The id field is checked for validity against the id of the TIP for which a TICS definition is in progress.

Example:

```
END  EE0F
```

Specifies the end of the definition of the TICS for TIP EE0F.

EXECUTE COMMAND FILE (XEC)

Schedules the execution of a DARTS Command File.

FORMAT:

XEC path [-E] [-A arg1... arg9]

ARGUMENTS:

path

Specifies the absolute pathname of the DARTS Command file to be executed.

-E

This optional argument causes the echo of each command as read from the file to the operator's console.

-A arg1...arg9

An optional argument specifying between one and nine substitution values, separated by spaces, used when processing individual commands contained in the DARTS Command file (path). These commands use the expression ?Pn (n= 1 to 9) in places where substitution is desired. The default argument is a null string.

NOTE

-A must be the last argument of the command string.

DESCRIPTION:

The execution of this command results in the sequential processing of every command in the specified file. Within the file each DARTS command mnemonic must begin in the first column of a line. An asterisk as the first character indicates a comment line.

NOTE

This command can only be issued in the interactive mode from the operator's console.

Example 1:

```
XEC  >UDD>TICS_EE  -E
```

Executes every command in the DARTS Command file named >UDD>TICS_EE, with echo of each command on the operator's console.

Example 2:

```
XEC >UDD>TICS_E0 -A $B4 $R4
```

Executes every command in the DARTS Command file named >UDD>TICS_E0, substituting $B4 and $R4 for the parameters ?P1 and ?P2, respectively, contained in the file >UDD>TICS_E0.

For example, if the commands in the file were:

```
DEF E001
REG ?P2 $R5
DAT $B1.?P2 ?P1 -L ARRAY
END E001
```

the effective commands would be:

```
DEF E001
REG $R4 $B5
DAT $B1.$R4 $B4 -L ARRAY
END E001
```

FORMAT LOGGED DATA (FMT)

Refer to Section 5 for a detailed description of this and other Formatter Control commands.

LOG

LOG FILE DEFINITION (LOG)

Establishes a new log file after closing out the current log file.

FORMAT:

```
LOG  path [-S size]
```

ARGUMENTS:

path

Specifies the absolute pathname of the log file to be established. The file is re-created each time it is specified in this command.

-S size

An optional argument specifying a log file of a number of blocks equal to the size parameter specified (in decimal). This file will be used in a wraparound mode by the DARTS buffer logger.

The actual size of each block in this file is governed by the size of the internal buffers defined by the BFR command.

DESCRIPTION:

This command must be preceded by the BFR command and must be issued before any ACT command. Omission of the -S argument results in the generation of an open-ended log file.

Example 1:

```
LOG  >SID>FIRST_LOG
```

Establishes an open-ended log file named >SID>FIRST_LOG

Example 2:

```
LOG >>UDD>LOG_FILE -S 500
```

Establishes a wraparound log file named >>UDD>LOG_FILE able to accommodate 500 buffer blocks.

REGISTER SET DEFINITION (REG)

Defines the register set whose contents is to be collected at the TIP in question.

FORMAT:

```
REG  list
```

ARGUMENT:

list

The list of registers must be separated by spaces and can be one or more of the following elements:

*	All registers
$R*	All $R registers
$Rx	Register $Rx (x=1-7)
$B*	All $B registers
$Bx	Register $Bx (x=1-7)
$I	Indicator register
$M1	Mode register M1

DESCRIPTION:

No register's content is collected when the REG command is omitted during the definition of a TICS. The REG command can appear only once in the definition and no register can be specified more than once in the command.

Compare its use in the definition of a TICS with those of the DEF, DAT, and END commands.

Example 1:

```
REG  $R1 $B4
```

Specifies collection of the contents of $R1 and $B4.

Example 2:

```
REG  $B* $I
```

Specifies collection of the contents of $I and all base registers.

Example 3:

```
REG  *
```

Specifies the collection of all R-registers, base registers, $I and $M1.

REMOVE TICS DEFINITION (REM)

Removes the definition of the specified list of TICS from the table of defined TICS.

FORMAT:

REM list

ARGUMENT:

list

A list of one to eight TIP id elements separated by spaces. Each TIP id element can be either a single id, including wildcards, or an id range specified as low/high.

DESCRIPTION:

Deletes the specified list of (active or inactive) TICS from the table of currently defined TICS. This command complements the DEF/END sequence of TICS definition commands. However, while TICS can only be defined individually, they can be removed as a list.

Refer to the ACT command description for more examples.

NOTE

Use REM only on defined TICS. For a TICS definition in progress refer to the ABT command.

EXAMPLE:

REM EE*

Removes the definitions of all TICS associated with TIP ids of module code EE.

RESET CONDITION FOR EXECUTION (RCX)

Resets the condition set (by the SCX command) for the execution of a specified TIP.

FORMAT:

RCX id

ARGUMENT:

id

A four-digit hexadecimal string that specifies a TIP id. The left pair of digits represents a module code and the right pair represents a site code in the module. 00 cannot be specified as a module code. (Refer to "TIP Identification Code" in this section.)

DESCRIPTION:

Resets an imposed condition for execution of a TIP, ensuring that the specified data is collected each time the Trace Point executes.

RESUME FORMATTING (RFM)

Refer to Section 5 for a detailed description of this and other Formatter Control commands.

RLG

RESUME LOGGING (RLG)

Resumes logging after a suspension.

FORMAT:

RLG

ARGUMENTS:

None

DESCRIPTION:

The RLG command can only be issued in the interactive mode. It displays a message at the console acknowledging execution of the command.

Example:

RLG

Resumes logging after logging has been suspended, and returns a message to the operator's console.

Message:

Logging Resumes

SET CONDITION FOR EXECUTION (SCX)

Sets the condition for execution of a specified TIP. Specify either a basic condition (Format 1) or a compound condition (Format 2).

FORMAT 1:

```
SCX id cond-1
```

FORMAT 2:

```
SCX id cond-1 oper-1 cond-2 [oper-2 cond-3]
```

ARGUMENTS:

id

A four-digit hexadecimal string specifying a TIP id. The left pair of digits represents a module code and the right pair represents a site code in the module. 00 cannot be specified as a module code. (Refer to "TIP Identification Code" in this section.)

cond-1 (cond-2 cond-3)

A conditional entity (without embedded blank separators) that has a logical value (.TRUE. or .FALSE.).

Specify the condition in one of two ways:

[^]<exp-1>[&<mask-1>]

or

[^]<exp-1>[&<mask-1>]<relation><exp-2>[&<mask-2>]

where:

<exp> can be a:

- Decimal literal
- Hexadecimal literal
- $R register
- Address syllable

<mask> is an optional hexadecimal literal

<relation> is one of six operators:

<	less than
=	equal to
>	greater than
^<	not less than
^=	not equal to
^>	not greater than

NOTE

See extended notes regarding the use of condition parameters in Notes for Condition Parameters below.

oper

A single character representing either a logical .AND. or a logical .OR. operator. Use it to combine up to three condition entities into a compound condition.

.AND. is represented by the ampersand ('&', X'26'), and .OR. is represented by the vertical bar ('|', X'7C'). The .AND. operator takes precedence over the .OR. operator.

DESCRIPTION:

Once a TICS is defined and activated, execution of a TIP results in complete data collection according to the TICS specification. The default is unconditional execution, everything will be gathered.

Imposing a condition on TIP execution ensures that data collection occurs only when the specified condition evaluates to a logical .TRUE. Conditional execution results either in no data being collected, or nothing but specified data being collected.

Example 1 (using Format 1):

```
SCX E204 $B2.6&X400
```

Specifies that data should be collected upon execution of TIP id E204 (only if bit 5) of the location at offset 6 of a structure pointed to by base register $B2, is set. Bit 5 is equivalent to a mask of X'0400'.

Example 2 (using Format 2):

```
SCX EF01 $R1^=0 & $B3.2&XFF00=X1500
```

Specifies that data should be collected upon execution of TIP id EF01 only if the content of $R1 is non-zero, and offset 2 of a structure pointed to by base register $B3 contains a value of 21 decimal (15 hexadecimal) in the left byte.

Notes for Condition Parameters

1. The optional leading NOT operator (^) is used to reverse the logical value of the evaluated condition.

2. A decimal literal is a signed value in the range of -32,768 to +32,767.

3. A hexidecimal literal is specified as a string of 1 to 4 hex-digits (leading zeros are assumed) prefixed by the character X. Do not use quotation marks.

4. A $R register should be specified as $Rn (n=1 to 7). Its contents will be used for the value of the <exp>.

5. The address syllable addresses a location whose contents is used as the value of the <exp>.

   The address syllable can be specified in one of four formats:

```
$Bx.Offset      (x=1 to 7)
*$Bx.Offset
$Bx.$Ry         (y=1 to 7)
*$Bx.$Ry
```

   The asterisk (*) represents indirect addressing. Offset is a signed decimal literal in the range of -32,768 and +32,767.

6. The mask cannot be specified if <exp> is a decimal or a hexadecimal literal.

   In the first format, <exp-1>&<mask-1> amounts to a Load-Bit instruction. The absence or presence of the leading NOT (^) operator turns the condition into a test for bit true or bit false.

   In the second format, the <mask> can be used to eliminate the "don't care" bit(s) in the value of an <exp> in order to specify conditions keyed by individual bytes.

SUSPEND LOGGING (SLG)

Suspends the logging facility.

FORMAT:

```
SLG
```

ARGUMENTS:

None

DESCRIPTION:

Logging can only be suspended; it cannot be terminated except when terminating DARTS with the OCL command:

```
$S DARTS -Q
```

Note that data collection ceases as soon as logging is suspended so as to avoid a buffer overrun condition.

This command can be issued only in the interactive mode. It displays a message at the console acknowledging execution of the command.

Example:

```
SLG
```

Suspends the logging facility, and displays a message at the operator's console.

Message:

```
Logging Suspended
```

SUSPEND FORMATTING (SFM)

Refer to Section 5 for a detailed description of this and other Formatter control commands.

TABLE OF ENTRIES FOR TICS (TOE)

Defines the size of the TICS table of entries.

FORMAT:

```
TOE  n
```

ARGUMENT:

n

Specifies the number of expected TICS entries.

DESCRIPTION:

This command must be issued before the first DEF command can begin a TICS definition.

The size of an existing table can be increased by issuing a TOE command with a larger value of n.

Example:

```
TOE  30
```

Specifies that up to 30 TICS definitions can follow.

TERMINATE FORMATTING (TFM)

Refer to Section 5 for a detailed description of this and other Formatter commands.

Section 5
LOG FILE FORMATTER

The log file formatter, scheduled by an interactive FMT command, is a spawned task operating at the same level as the DARTS command manager. Subsequent user commands can be used to control the formatter request.

Invoking DARTS from any user task group results in its exclusive use as a formatter. In this mode the DARTS formatter cannot be controlled; it either executes to completion or aborts when an error occurs. The command requires arguments that constitute a formatter command resulting in the execution of a single formatter request.

The format of the ECL command is:

```
DARTS FMT path_name argument_list
```

where:

path_name

Specifies the absolute pathname of the log file being formatted.

argument_list

Refer to FMT command in this section.

LOG FORMATTER OUTPUT

Formatted output has a standard format. For each TIP there is:

- A time-stamped header line
- A TIP id in hexadecimal notation
- The sequence number of the TIP's execution
- The $R and $B register contents, when selected.

Refer to Appendix E for a complete example and description of formatted output.

FORMATTER COMMANDS

The remainder of this section provides a detailed description of the Formatter commands. They are presented in alphabetical order.

FORMAT LOGGED DATA (FMT)

Generates formatted output from the raw data in the specified log file.

FORMAT:

```
FMT {path} [-O output_file] [-F date_time] [-T date_time]
    {-L }
                [ -I list]
```

ARGUMENTS:

path

Specifies the log file to be formatted. If the current log file is to be formatted, use the keyword -L instead to select online current formatting.

-O output_file

Directs formatted output to an output_file that is re-created each time.

Default: !LPT00

-F date_time

Specifies the leading edge of the time window. Date_time must be specified in the form yyyy/mm/dd:hhmm or hhmm, with the current date being used in the latter case.

Default: The time at the beginning of the log file.

-T date_time

Specifies the trailing edge of the time window. The format of date_time is as described above.

Default: The time at the end of the log file.

-I list

Specifies a list of one to eight TIP id elements, separated by spaces. If omitted, all TIPs in the time window are formatted. Each TIP id element can be either a single id, including wildcards, or an id range specified as low/high. Refer to the description of the ACT command (refer to Section 4) for details on TIP id element specification.

DESCRIPTION:

The selection of information to be formatted is undertaken in the following manner:

1. A time window is specified by the -F and -T arguments. This determines the absolute perimeter of the formatting operation.

2. In a window, either all the TIPs or a select list of TIPs can be formatted.

This command can be issued only in the interactive mode.

Example:

```
FMT >SID>FIRST_LOG -O !LPT00 -I EF* EC01/EC07
    -F 1986/2/1:1431 -T 1986/2/1:1435
```

Generates formatted output from the data in the log file >SID>FIRST_LOG. Directs output to the line printer LPT00. The time window for formatting is 2:31pm to 2:35pm on February 1, 1986. Displays only the data captured on behalf of the module code EF and all TIPs in the range EC01 and EC07.

RESUME FORMATTING (RFM)

Resumes the execution of the suspended formatter.

FORMAT:

RFM

ARGUMENTS:

None.

DESCRIPTION:

This command can only be issued in the interactive mode. It displays a message at the console acknowledging execution of the command.

Example:

RFM

Resumes execution of a formatting request and displays a message at the operator's console.

Message:

Formatting Resumes

SUSPEND FORMATTING (SFM)

Suspends the execution of the formatter.

FORMAT:

SFM

ARGUMENTS:

None.

DESCRIPTION:

This command can only be issued in the interactive mode. It displays a message at the console acknowledging execution of the command.

Example:

SFM

Suspends execution of a formatting request, and displays a message at the operator's console.

Message:

Formatting Suspended

TERMINATE FORMATTER (TFM)

Terminates the execution of a formatter request.

FORMAT:

TFM

ARGUMENTS:

None.

DESCRIPTION:

The TFM command is the sole means of termination when using the formatter to process the online log file (even as the data is being logged) with no specified time window.

The TFM command can be issued only in the interactive mode. It displays a message at the console acknowledging execution of the command.

Example:

TFM

The formatter is terminated with indication at the console, and displays a message at the operator's console.

Message:

Formatting Request Terminating

Appendix A
ERROR MESSAGES

The MOD 400 message library maintains error messages (and some informational messages) issued by DARTS. The DARTS component code is 6F; therefore, all messages use the code 6Fxx.

INTERPRETING MESSAGES

All messages use the character string substitution facility to identify the subcomponent of DARTS that is issuing the message and other pertinent information. For example, the error message issued when processing the command DEF 0X23 is as follows:

```
6F36  DEF:  Invalid TIP id:  0X23.  It should be precisely
            4 hex digits.
```

where:

6F36

6F is the DARTS component code, and 36 is the message identification number.

DEF

The DARTS subcomponent name.

0X23

The invalid TIP id in this example. It contains a nonhexadecimal digit.

LIST OF ERROR AND INFORMATION MESSAGES

As a way of representing actual entries, these messages adhere to the following convention:

- A lowercase letter enclosed in a less than (<) sign and greater than (>) sign represents a returned string. For example, <pppp> and <iiiii> represent returned strings.
- xxx represents a command mnemonic.

6F03 xxx:	DARTS is not configured in the system. Include LDBU ZXDJ directive in CLM_USER file and reboot.
6F04 xxx:	DARTS monitor is already in use. Duplicate invocation rejected.
6F05 xxx:	DARTS is still executing previously issued command. This command is being ignored.
6F06 xxx:	DARTS monitor operations are being terminated.
6F07 xxx:	This command cannot be issued before invoking DARTS monitor.
6F08 xxx:	DARTS invocation arguments can only be: -D <path_name> and -E, with -A as the last keyword.
6F09 xxx:	Data base (pathname) error: <pppp>.
6F0A xxx:	Initialization overlay could not be loaded.
6F0B xxx:	Exit processor overlay could not be loaded.
6F0C xxx:	XEC command processor overlay could not be loaded.
6F0D xxx:	Command processor overlay could not be loaded for <xxx> command.
6F0E xxx:	Formatter overlay could not be loaded.
6F0F xxx:	DARTS command execution has completed.
6F10 xxx:	Required memory block could not be obtained.
6F11 xxx:	DARTS initialiaation in progress. Please wait for the completion message.

6F12 xxx: DARTS initialization completed. Monitor is ready for commands.

6F13 xxx: DARTS initialization has failed. Please correct the error and retry.

6F15 xxx: Log file switch is not possible in case of circular log files.

6F16 xxx: Error in log file size specification via -S <n> option. <n> should range from 16 to 4096.

6F17 xxx: Get file information error for the file: <pppp>.

6F18 xxx: Create file error for the file: <pppp>.

6F19 xxx: Get file error for the file: <pppp>.

6F1A xxx: Open file error for the file: <pppp>.

6F1B xxx: Read record error for the file: <pppp>.

6F1C xxx: Write record error for the file: <pppp>.

6F1D xxx: Read block error for the file: <pppp>.

6F1E xxx: Write block error for the file: <pppp>.

6F21 xxx: Unbalanced quotes in command input: <iiiiii>.

6F22 xxx: Invalid command mnemonic: <cccc>.

6F23 xxx: Unknown/unsupported command: <cccc>.

6F24 xxx: Required argument(s) missing. Issue this command without arguments to determine its format.

6F25 xxx: Excessive arguments specified. Issue this command without arguments to determine its format.

6F26 xxx: Unknown argument/keyword specified in command input: <kkkk>.

6F27 xxx: Command permitted only in interactive mode: <cccc>.

6F28 xxx: TOE command must be issued before this command.

6F29 xxx: LOG command must be issued before this command.

6F2A xxx: BFR command must be issued before this command.

6F2B xxx: No log file assigned as yet.

6F2C xxx: Log file switch rejected since concurrent formatting is in progress.

6F2D xxx: No formatting request in progress.

6F2E xxx: Formatting request is already in progress.

6F30 xxx: Specified log file is not a DARTS created log file: <pppp>.

6F31 xxx: TICS definition is already in progress. END the current definition before attempting a new one.

6F32 xxx: TICS definition is not in progress. This command is being ignored.

6F33 xxx: TICS definition is being duplicated for Id: <hhhh>. Remove the old definition before attempting redefintion.

6F34 xxx: Undefined TIP id: <hhhh>.

6F35 xxx: The specified TIP id: <hhhh> does not match the id currently being defined: <hhhh>.

6F36 xxx: Invalid TIP id: <hhhh>. It should be precisely four hexadecimal digits.

6F37 xxx: Invalid TIP id: <hhhh>. The error is in the module/site code.

6F38 xxx: Invalid TIP id delimiter.

6F39 xxx: Data entry overflow (>4) in current TICS definition. This definition should be ended (END) or aborted (ABT).

6F3A xxx: Invalid/unsupported address field in DAT entry: <aaaa>.

6F3B xxx: Invalid base register specification: <rrrr>.

6F3C xxx: Invalid index register specification: <rrrr>.

6F3D xxx: Invalid range register specification: <rrrr>.

6F3E xxx: Invalid offset specification: <oooo>.

6F3F xxx: Invalid range literal specification: <llll>.

6F40 xxx: Range (literal) specification exceeds prescribed limit (2000 bytes).

6F41 xxx: Data entry Label specification is missing.

6F42 xxx: Invalid delimiter to base/index register.

6F43 xxx: Range limit (-M) can be specified only for dynamic ranging.

6F44 xxx: Range limit (-M) specification is in error.

6F45 xxx: Range limit (-M) specification exceeds prescribed limit (2000 bytes).

6F4A xxx: Duplicate REG command issued in this TICS definition.

6F4B xxx: Invalid register specification: <rrrr>.

6F4C xxx: Duplication of register specification with the selection: <rrrr>.

6F4F xxx: Buffer low condition corrected. Currently <nnnnn> TIPs have executed for which DARTS was unable to collect data due to buffer low condition.

6F50 xxx: Three arguments are required with the BFR command.

6F51 xxx: BFR arguments have already been defined. This command is being rejected.

6F52 xxx: Buffer size argument error: <nnnnn>.

6F53 xxx: Buffer initial count argument error: <nnnnn>.

6F54 xxx: Buffer low threshold argument error: <nnnnn>.

6F55 xxx: The BFR initial count argument has to exceed the low threshold argument.

6F56 xxx: TICS Table of Entries (TICTOE) overflow condition. Expand the table with a new TOE command before attempting new definitions.

6F57 xxx: Invalid value for the size of the Table of Entries (TOE): <nnnnn>.

6F58 xxx: Specified size value cannot be less than or equal to the existing value. The TICTOE table can only be expanded.

6F59 xxx: Log File pathname must be specified in command input.

6F5A xxx: Output File pathname has not been specified in command input.

6F5B xxx: Specified value is in error for the keyword: <kkkk>.

6F5C xxx: Formatter Time Window specification or mapping error.

6F5D xxx: No more than eight TIP id elements are allowed with -I keyword.

6F5E xxx: Invalid combination of formatter options results from the selection of the keyword: <kkkk>.

6F5F xxx: Duplicate specification of the keyword: <kkkk>.

6F60 xxx: Required argument is missing for the keyword: <kkkk>.

6F72 xxx: Slewing through command file until END directive encountered (or end of file).

6F73 xxx: Slewing stops with the command: <iiiiii>.

6F74 xxx: Only -E and -A are allowed as keyword arguments.

6F75 xxx: More than nine arguments follow the -A keyword. The -A should be the last keyword in the command.

6F76 xxx: No argument specified after the -A keyword.

6F81 xxx: No condition has been specified in this command.

6F82 xxx: Too many conditions have been specified in the command. Only up to three basic conditions, suitably compounded, can be specified.

6F83 xxx: The compound condition specified is not properly structured. The compounding operators can only be & (AND) or | (OR).

6F84 xxx: Invalid condition compounding specified in the command: <ccccc>.

6F85 xxx: Incomplete condition expression specified: <eeeee>. The error is at about byte offset: <bbbb>.

6F86 xxx: Invalid logical NOT operator specified: <eeeee>. The error is at about byte offset: <bbbb>.

6F87 xxx: Invalid indirection operator (*) specified: <eeeee>. The error is at about byte offset: <bbbb>.

6F88 xxx: Indirection can be used only with $Bx addressing: <eeeee>. The error is at about byte offset: <bbbb>.

6F89 xxx: A valid register mnemonic does not follow $ sign: <eeeee>. The error is at about byte offset: <bbbb>.

6F8A xxx: Invalid $Rx register specified: <eeeee>. The error is at about byte offset: <bbbb>.

6F8B xxx: Invalid $Bx register specified: <eeeee>. The error is at about byte offset: <bbbb>.

6F8C xxx: Invalid extension specified after $Bx: <eeeee>. The error is at about byte offset: <bbbb>.

6F8D xxx: Invalid index register specified: <eeeee>. The error is at about byte offset: <bbbb>.

6F8E xxx: Invalid hexadecimal string specified: <eeeee>. The error is at about byte offset: <bbbb>.

6F8F xxx: Presumed decimal number is not specified: <eeeee>. The error is at about byte offset: <bbbb>.

6F90 xxx: Invalid AND mask specified: <eeeee>. The error is at about byte offset: <bbbb>.

6F91 xxx: Invalid relation operator specified: <eeeee>. The error is at about byte offset: <bbbb>.

6F92 xxx: Required expression following a relation operator is missing: <eeeee>. The error is at about byte offset: <bbbb>.

6F96 xxx: A condition is already defined for the TIP id: <hhhh>. Reset the condition with an RCX command and retry.

Appendix B
DATA BASE RECOMMENDATIONS

Include the following three commands in the initialization data base file in the order presented:

TOE n

Defines the size of the TIP Table of Entries (TIPTOE). (n should be greater than the number of TICS likely to be specified in subsequent DEF commands. Refer to the TOE description in Section 4.)

BFR size number low_limit

To define the chain of internal buffers to be used by the DARTS Data Collector. Refer to the BFR description in Section 4.

LOG path_name

To define the initial log file to be used. Unless defined as part of the initialization process, the commands issued to DARTS from the data base are not logged. Refer to the LOG description in Section 4.

Sample Contents of >>SID>BULLSEYE

```
TOE  30
BFR  1000 8 4
LOG  >UDD>DARTS_LOG
```

Appendix C
USING DARTS: AN EXAMPLE

Figure C-1 shows a software module called TEST. The module contains predefined Trace Points EE01, EE02, and EE03. DARTS traced TEST's execution using the Trace Collection Specifications (TICS) set forth in Figure C-2. Figure C-3 illustrates formatted trace data. TEST was executed four times; each time using a different OCL command format.

The TEST program shows the use of both the $tip and $tipsp macros. $tipsp generates an environment for special interface TIPs EE02 and EE03. TEST executed in the system pool ($$), allowing it use of the special interface. If executed from any other pool, TIPs EE02 and EE03 would have amounted to no-ops, due to lack of ring 0 privilege.

The TEST program was located under both the HIS and SYSLIB1 directories. The four OCL commands used in the test were:

```
$S >HIS>TEST PARM-ONE
$S >HIS>TEST
$S >SYSLIB1>TEST PARM-ONE-IS-MORE-THAN-10-BYTES
$S >SYSLIB1>TEST TERRY BENTLEY
```

```
000290              0000            TEST      EQU      $
000291                              *
000292                              ***  ENTERED  WITH $B4 --> TRB AND $B7 --> PARM BLOCK
000293                              *
000294  0000  9877                          LDR      $R1,+$B7          $R1 <- COUNT OF PARMS (INCL BU NAME)
000295  0001  ACF7                          LDB      $B2,+$B7          $B2 <- PTR TO BU NAME
000296  0002  A872                          LDR      $R2,+$B2          $R2 <- ITS LENGTH (IN BYTES)
000297  0003  BCF7                          LDB      $B3,+$B7          $B3 <- PTR TO 1STARG (OR NULL)
000298  0004  B873                          LDR      $R3,+$B3          $R3 <- ITS LENGTH (OR X'83C0')
000299                              *
000300                              ***  RELEVENT REGISTER SET IS: $R1-$R3 AND $B2-$B4 AND $B7.
000301                              *
000302                              @         $TIP     EE01              GENERAL INTERFACE TIP
000303  0005  1234                          DC       Z'1234'           DARTS INSTRUCTION
000304  0006  EE01                          DC       Z'EE01'           TIP ID
000305                              *
000306                              @         $TIPSP   $TIP WILLNOW REVERT TO SPECIAL INTERFACE GENERATION
000307                              ***
000308                              ***       ONLY SPECIAL INTERFACE TIPS BEING DEFINED HENCEFORTH
000309                              ***
000310  0007  9CD7                          LDB      $B1,=$B7          $B1 <- PTR TO REST OF PARM BLOCK
000311  0008  FBC0 0056                     LAB      $B7,STACK         $B7 <- STACK REQUIRED BY SPECIAL INTERFACE
000312  000A  B970 83C0                     CMR      $R3,=Z'83C0'      IF IT IS THE END OF THE ROAD...
000313  000C  0903                          BE       >SKIP
000314                              *                                   $B3 <- PTR TO FIRST ARGUMENT AND
000315                              *                                   $R3 <- ITS LENGHT (IN BYTES)
000316  000D  CCF1                          LDB      $B4,+$B1          $B4 <- PTR TO 2ND ARG, OR NULL
000317  000E  C874                          LDR      $R4,+$B4          $R4 <- ITS LENTHG (OR X'83C0')
000318                              *
000319              000F            SKIP      EQU      $
000320                              *
000321                              ***  RELEVENT REGISTER SET MAY BE: $R1-$R4 AND $B1-$B4.
000322                              *
000323                              @         $TIP     EE02              SPECIAL INTERFACE TIP EXAMPLE 1
000324  000F  DC80 0000 0018X               LDB      $B5,<ZCHOMM+X'18'  $B5 <- SCB PTR
000325  0012  D3CD 0102                     LNJ      $B5,*$B5.S_DART   CALL TO DARTS
000326  0014  EE02                          DC       Z'EE02'           TIP ID
000327                              *
000328  0015  EC80 0000 0018X               LDB      $B6,<ZHCOMM+X'18'  $B6 <- SCB PTR (SET UP NEXT $TIP)
000329                              *
000330                              @         $TIP     EE03,S_DART       SPECIAL INTERFACE TIP EXAMPLE 2
000331  0018  D3CE 0102                     LNJ      $B5,*$B6.S_DART   CALL TO DARTS
000332  001A  EE03                          DC       Z'EE03'           TIP ID
000333                              *
000334                              @         $TRMRQ   =0                TERMINATE THE TASK
000335  001B  AB70 0000                     LDR      $R2,=0            SET $R2 TO COMPLETION STATUS
000336  001D  0001                          MCL
000337  001E  0103                          DC       X'0103'           TERMINATE WITHOUT MODIFYING START ADDRESS
000338                              *
000339  001F        5454                    RESV     64,A'TT'
000340              005F            STACK     EQU      $                 THE STACK
000341  005F  5454                          RESV     1,A'TT'
```

Figure C-1. TEST: A Software Module With Predefined Trace Points

```
*
* TIP ID EE01 COLLECTS TRB AND PRMBLK INFO
*
DEF EE01
REG $R1 $R2 $R3 $B2 $B3 $B4 $B7
DAT $B4 12 -L TRB
DAT $B7.-5 60 -L PRMBLK
END EE01
*
* TIP ID EE02 REPORTS ARG-1 IF IT EXISTS
*
DEF EE02
DAT $B3 $B3,-1 -L ARG-1 -M 10
DAT $B3 $R3.-1 -L SAM_ARG
END EE02
*
* IMPOSE CONDITION TO INSURE IT DOES NOT EXECUTE WITHOUT ARG-1
*
SCX EE02 $R1^=1
*
* TIP ID EE03 REPORTS BU_NAME
*
DEF EE03
DAT $B2 $R2 -L BU_NAME
END EE03
*
* NOW ACTIVATE ALL 3
*
ACT EE*
```

Figure C-2. Trace Collection Specifications (TICS) for TIPs EE01, EE02, and EE03.

A SPECIFIC TICS EXAMPLE

The TICS specification statements for TIP EE02 (Figure C-2) are of some interest. The user context at that point for TIP EE02, includes, in the $B3/$R3 register pair, the pointer to, and the range of the first argument in the OCL command, if it exists. The contents of the $R3 register equal those of $B3.-1. The two DAT arrays use dynamic ranging to address the same memory. The first array is subject to a 10-byte limit. The second array has an implicit 2000-byte limit. The imposed condition (SCX) set for TIP EE02 ensures that it will not collect data unless the number of parameters in the $R1 register is other than one.

FORMATTED OUTPUT

For each TIP, the output has a standard format (Figure C-3). The header line indicates the TIP id (in hexadecimal notation), the sequence number of the TIP's execution (in decimal notation), and the time stamp.

The next two lines display the $R and $B registers contents. These lines are absent when no register was selected in the TICS definition.

All selected data array entries at the TIP are output in sequence on the succeeding lines. If no array entry was selected in the TICS definition, the formatted output for the TIP is complete at this stage. The entry number (1 through 4) and its label, if any, are at the left margin of the first line of output. Up to 32 bytes of the collected data array appear in hexadecimal notation. The right margin displays the ASCII character equivalent of the data. Multiple lines of output result when the captured data exceeds 32 bytes of information.

In Figure C-3, the absence of TIP EE02's output, between EE01 and EE03 (sequence 2), for the second execution demonstrates the suppression of data collection due to the imposed condition evaluating to .FALSE. In the next sequence (third execution), TIP EE02's output displays the truncation of the first DAT array brought about by setting the -M limit on data collection.

Note that in the third execution, TIP EE02 has a sequence label of 0002; TIP's EE01 and EE03 have sequence labels of 0003.

```
TIP:EE01  SEQ:0001                TIME: 1985/07/19 1200:56.500
          $B7: 21F8F  $B4: 21F84  $B3: 21F9B  $B2: 21F92
          $R3:  0008  $R2:  0009  $R1:  0002
1)  TRB       0000 8DE4 00C1 FF01 CCCC CCCC                                    ...D.A..LLLL
2)  PRMBLK    0002 0002 1F91 0002 1F97 0000 0000 0009 3E48 4953 3E54 4553 5420 0008 5041 524D    .............>HIS>TEST ..PARM
              2D4F 4E45 2020 CCCC CCCC CCCC CCCC CCCC CCCC CCCC CCCC CCCC CCCC  -ONE  LLLLLLLLLLLLLLLLLLLL

TIP:EE02  SEQ:0001                TIME: 1985/07/19 1200:56.500
1)  ARG-1     5041 524D 2D4F 4E45                                              PARM-ONE
2)  SAM_ARG   5041 524D 2D4F 4E45                                              PARM-ONE

TIP:EE03  SEQ:0001                TIME: 1985/07/19 1200:56.500
1)  BU_NAME   3E48 4953 3E54 4553 5420                                         >HIS>TEST

TIP:EE01  SEQ:0002                TIME: 1985/07/19 1201:03.500
          $B7: 21FAF  $B4: 21FA4  $B3: 00001  $B2: 21FB0
          $R3:  83C0  $R2:  0009  $R1:  0001
1)  TRB       0000 8DCC 00C1 FF01 CCCC CCCC                                    ...L.A..LLLL
2)  PRMBLK    0001 0002 1FAF 0000 0000 0009 3E48 4953 3E54 4553 5420 0102 0002 1FA4 0002 20E4    ...../......>HIS>TEST .....$.. D
              CCCC CCCC CCCC CCCC CCCC CCCC 0006 0000 0000 0000 CCCC CCCC CCCC CCCC  LLLLLLLLLLLL........LLLLLLLL

TIP:EE03  SEQ:0002                TIME: 1985/07/19 1201:03.500
1)  BU_NAME   3E48 4953 3E54 4553 5420                                         >HIS>TEST

TIP:EE01  SEQ:0003                TIME: 1985/07/19 1201:25.850
          $B7: 21F8F  $B4: 21F84  $B3: 21F9A  $B2: 21F92
          $R3:  001E  $R2:  000D  $R1:  0002
1)  TRB       0000 8DFC 00C1 FF01 CCCC CCCC                                    ...\.A..LLLL
2)  PRMBLK    0002 0002 1F91 0002 1F99 0000 0000 000D 3E53 5953 4C49 4231 3E54 4553 5420 001E    .............>SYSLIB1>TEST ..
              5041 524D 2D4F 4E45 2D49 532D 4D4F 5245 2D54 4841 4E2D 3130 2D42 5954  PARM-ONE-IS-MORE-THAN-10-BYT

TIP:EE02  SEQ:0002                TIME: 1985/07/19 1201:25.900
1)  ARG-1     5041 524D 2D4F 4E45 2D49                                         PARM-ONE-I
2)  SAM_ARG   5041 524D 2D4F 4E45 2D49 532D 4D4F 5245 2D54 4841 4E2D 3130 2D42 5954 4553  PARM-ONE-IS-MORE-THAN-10-BYTES

TIP:EE03  SEQ:0003                TIME: 1985/07/19 1201:25.900
1)  BU_NAME   3E53 5953 4C49 4231 3E54 4553 5420                               >SYSLIB1>TEST

TIP:EE01  SEQ:0004                TIME: 1985/07/19 1201:57.450
          $B7: 21F8F  $B4: 21F84  $B3: 21F9C $B2:  21F94
          $R3:  0005 $R2:  000D  $R1:0003
1)  TRB       0000 0760 00C1 FF01 CCCC CCCC                                    ...@.A..LLLL
2)  PRMBLK    0003 0002 1F93 0002 1F9B 0002 1F9F 0000 0000 0000 3E53 5953 4C49 4231 3E54 4553    ...................>SYSLIB1>TES
              5420 0005 5445 5252 5920 0007 4245 4E54 4C45 5920 0102 0002 1F84 0002  T ..TERRY ..BENTLEY ........

TIP:EE02  SEQ:0003                TIME: 1985/07/19 1201:57.500
1)  ARG-1     5445 5252 5920                                                   TERRY
2)  SAM_ARG   5445 5252 5920                                                   TERRY

TIP:EE03  SEQ:0004                TIME: 1985/07/19 1201:57.500
1)  BU_NAME   3E53 5953 4C49 4231 3E54 4553 5420                               >SYSLIB1>TEST

TIP:0007  SEQ:0001                TIME: 1985/07/19 1202:05.850
```

Figure C-3. Collected Trace Data as Formatted by the Log File Formatter.

GLOSSARY

Active TIP

An explicitly activated TIP that is capable of collecting data when encountered (executed).

Buffers

Internal Buffers.

Buffer Logger

Nondistributed component of DARTS that outputs buffers (passed from the data collector) to the online Log File.

Buffer Manager

Nondistributed component of DARTS that maintains the chain of internal buffers. The task operates in tandem with the buffer logger.

Command

Issued at initialization (from the data base file) or subsequently by the operator to control the execution of DARTS. Some commands can be issued only by the operator.

Command Manager

Component of DARTS that accepts and parses interactive input from the operator and schedules the appropriate command processor.

Command Processor

Component of DARTS that performs the action of a particular command. Scheduled from either the command manager or the command file processor.

Data Collector

Distributed component of DARTS that is responsible for data collection at a TIP. It passes internal buffers (filled with the data collected at various TIPs) to the buffer logger.

Formatter

Log file formatter.

General Interface

One of two ways to define a TIP, by using the trace instruction to generate a trap that in turn leads to the data collector. Available to both system level and user level software modules.

Inactive TIP

The initial state of a TIP, when it is incapable of collecting data. A TIP has to be explicitly activated for use.

Internal Buffers

Memory blocks of specified size and (initial) number, maintained in a chain, that serve as the recording medium for the data collector.

Link-and-Jump Interface

Special Interface.

Log File

An online disk file to which the buffer logger outputs buffers filled with collecting data.

Log File Formatter

Spawned task component of DARTS that processes the information in a log file (online or offline).

Logger

Buffer logger.

Module Code

The left pair of hexadecimal digits in a 4-digit hexadecimal TIP id. A unique code that is preassigned for each software module, with different TIPs in a module having different site codes. The Module code of 00 is reserved for the exclusive use of the DARTS facility.

Pre-defined TIP

A permanent TIP that is assembled inline by the developer of a software module, to provide a standard data collection site in released software.

Site Code

The right pair of hexadecimal digits in a 4-digit hexadecimal TIP id. Used to identify a TIP definition in a specific module.

Special Interface

One of two ways to define a TIP, with a link-and-jump call to the data collector. Available only to system level modules.

SCB

System Control Block.

Trace Information Capture Specification

Structure that defines the user register context and/or user data arrays to be collected at a TIP. This structure is under the control of DARTS and is external to the software in which the TIP is defined.

Trace Information Point

A calling sequence in the user software that marks the site for information collected.

Trap Interface

General interface.

Trap Processor

Special routine to process the trap from the trace instruction that serves as the interface to the data collector.

TICS

Trace Information Capture Specification.

TIP

Trace Information Point.

TIP id

A 4-digit hexadecimal (16-bit) number that uniquely identifies a TIP. The left pair of digits is interpreted as the module code, and the right pair is interpreted as the site code.

User Data Array

An array of data words/bytes that is collected at a TIP according to the specification in the TICS structure. The specification is represented by an entry in the TICS structure.

User Register Context

The execution context of the DARTS user, preserved across the call to the trace facility that can itself be collected and/or used in data collection at a TIP.

INDEX

CUT ALONG

HONEYWELL INFORMATION SYSTEMS
Technical Publications Remarks Form

TITLE: DPS 6 GCOS 6
DATA BASE AUGMENTED
REAL-TIME TRACING SYSTEM
USER'S GUIDE

ORDER NO. CZ74-01

DATED MARCH 1986

ERRORS IN PUBLICATION

SUGGESTIONS FOR IMPROVEMENT TO PUBLICATION

Your comments will be investigated by appropriate technical personnel and action will be taken as required. Receipt of all forms will be acknowledged; however, if you require a detailed reply, check here. ☐

FROM: NAME _____ DATE _____

TITLE _____

COMPANY _____

ADDRESS _____

_____

PLEASE FOLD AND TAPE—
NOTE: U. S. Postal Service will not deliver stapled forms

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 39531 WALTHAM, MA02154

POSTAGE WILL BE PAID BY ADDRESSEE

HONEYWELL INFORMATION SYSTEMS
200 SMITH STREET
WALTHAM, MA 02154

ATTN: PUBLICATIONS, MS486

Honeywell

HONEYWELL INFORMATION SYSTEMS
Technical Publications Remarks Form

TITLE: DPS 6 GCOS 6
DATA BASE AUGMENTED
REAL-TIME TRACING SYSTEM
USER'S GUIDE

ORDER NO. CZ74-01

DATED MARCH 1986

ERRORS IN PUBLICATION

SUGGESTIONS FOR IMPROVEMENT TO PUBLICATION

Your comments will be investigated by appropriate technical personnel and action will be taken as required. Receipt of all forms will be acknowledged; however, if you require a detailed reply, check here. ☐

FROM: NAME _____ DATE _____

TITLE _____

COMPANY _____

ADDRESS _____

_____

PLEASE FOLD AND TAPE—
NOTE: U. S. Postal Service will not deliver stapled forms

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 39531 WALTHAM, MA02154

POSTAGE WILL BE PAID BY ADDRESSEE

HONEYWELL INFORMATION SYSTEMS
200 SMITH STREET
WALTHAM, MA 02154

ATTN: PUBLICATIONS, MS486

Honeywell

HONEYWELL INFORMATION SYSTEMS
Technical Publications Remarks Form

TITLE: DPS 6 GCOS 6
DATA BASE AUGMENTED
REAL-TIME TRACING SYSTEM
USER'S GUIDE

ORDER NO. CZ74-01

DATED MARCH 1986

ERRORS IN PUBLICATION

SUGGESTIONS FOR IMPROVEMENT TO PUBLICATION

Your comments will be investigated by appropriate technical personnel and action will be taken as required. Receipt of all forms will be acknowledged; however, if you require a detailed reply, check here. ☐

FROM: NAME _____ DATE _____

TITLE _____

COMPANY _____

ADDRESS _____

_____

PLEASE FOLD AND TAPE–
NOTE: U. S. Postal Service will not deliver stapled forms

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 39531 WALTHAM, MA02154

POSTAGE WILL BE PAID BY ADDRESSEE

HONEYWELL INFORMATION SYSTEMS
200 SMITH STREET
WALTHAM, MA 02154

ATTN: PUBLICATIONS, MS486

Honeywell