297-1001-533

DMS-100 Family

Bit Error Rate Performance Testing

Testing Reference

TELECOM10 Standard 07.01 April 1999

NORTEL
NORTHERN TELECOM

DMS-100 Family

Bit Error Rate Performance Testing

Testing Reference

Publication number: 297-1001-533
Product release: TELECOM10
Document release: Standard 07.01
Date: April 1999

Printed in the United States of America

NORTHERN TELECOM CONFIDENTIAL: The information contained in this document is the property of Northern Telecom. Except as specifically authorized in writing by Northern Telecom, the holder of this document shall keep the information contained herein confidential and shall protect same in whole or in part from disclosure and dissemination to third parties and use same for evaluation, operation, and maintenance purposes only.

Information is subject to change without notice. Northern Telecom reserves the right to make changes in design or components as progress in engineering and manufacturing may warrant.
This equipment has been tested and found to comply with the limits for a Class A digital device pursuant to Part 15 of the FCC Rules, and the radio interference regulations of the Canadian Department of Communications. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference in which case the user will be required to correct the interference at the user's own expense
Allowing this equipment to be operated in such a manner as to not provide for proper answer supervision is a violation of Part 68 of FCC Rules, Docket No. 89-114, 55FR46066
The SL-100 system is certified by the Canadian Standards Association (CSA) with the Nationally Recognized Testing Laboratory (NRTL).
This equipment is capable of providing users with access to interstate providers of operator services through the use of equal access codes. Modifications by aggregators to alter these capabilities is a violation of the Telephone Operator Consumer Service Improvement Act of 1990 and Part 68 of the FCC Rules

Publication history

April 1999

Standard 07.01. For the define command, add parameter DGRP for LCME lens and parameter ISLC for ISDN line cards. SR 40049769.

April 1998

Standard 06.02. Document migrated into new template.

Contents

About this document

When to use this document

The information contained in this publication applies to offices with Batch Change Supplement release 28 (BCS28) software. The editions for a given BCS release are in practice refer to *Product Documentation Directory*, 297-8991-001.

How to check the version and issue of this document

The version and issue of the document are indicated by numbers, for example, 01.01.

The first two digits indicate the version. The version number increases each time the document is updated to support a new software release. For example, the first release of a document is 01.01. In the *next* software release cycle, the first release of the same document is 02.01.

The second two digits indicate the issue. The issue number increases each time the document is revised but rereleased in the *same* software release cycle. For example, the second release of a document in the same software release cycle is 01.02.

To determine which version of this document applies to the software in your office and how documentation for your product is organized, check the release information in *DMS-100 Family Guide to Northern Telecom Publications*, 297-1001-001.

References in this document

The following documents are referred to in this document:

- *Product Documentation Directory*, 297-8991-001
- *Provisioning Guide*, PLN-8991-104 (section 450/32)
- *Office Feature Record*, D-90
- *SuperNode Technical Specification*, PLN-5001-001
- *Translations Guide*, 297-YYYY-350
- *Networks Maintenance Guide*, 297-1001-591

- *Trunks Maintenance Guide*, 297-1001-595
- *Maintenance System Man-Machine Interface Description*, 297-1001-520
- *Lines Maintenance Guide*, 297-1001-594

What precautionary messages mean

The types of precautionary messages used in NT documents include attention boxes and danger, warning, and caution messages.

An attention box identifies information that is necessary for the proper performance of a procedure or task or the correct interpretation of information or data. Danger, warning, and caution messages indicate possible risks.

Examples of the precautionary messages follow.

ATTENTION Information needed to perform a task

> **ATTENTION**
>
> If the unused DS-3 ports are not deprovisioned before a DS-1/VT Mapper is installed, the DS-1 traffic will not be carried through the DS-1/VT Mapper, even though the DS-1/VT Mapper is properly provisioned.

DANGER Possibility of personal injury

> **DANGER**
>
> **Risk of electrocution**
>
> Do not open the front panel of the inverter unless fuses F1, F2, and F3 have been removed. The inverter contains high-voltage lines. Until the fuses are removed, the high-voltage lines are active, and you risk being electrocuted.

WARNING Possibility of equipment damage

WARNING
Damage to the backplane connector pins
Align the card before seating it, to avoid bending the backplane connector pins. Use light thumb pressure to align the card with the connectors. Next, use the levers on the card to seat the card into the connectors.

CAUTION Possibility of service interruption or degradation

CAUTION
Possible loss of service
Before continuing, confirm that you are removing the card from the inactive unit of the peripheral module. Subscriber service will be lost if you remove a card from the active unit.

How commands, parameters, and responses are represented

Commands, parameters, and responses in this document conform to the following conventions.

Input prompt (>)

An input prompt (>) indicates that the information that follows is a command:

>BSY

Commands and fixed parameters

Commands and fixed parameters that are entered at a MAP terminal are shown in uppercase letters:

>BSY CTRL

Variables

Variables are shown in lowercase letters:

>BSY CTRL ctrl_no

The letters or numbers that the variable represents must be entered. Each variable is explained in a list that follows the command string.

Responses

Responses correspond to the MAP display and are shown in a different type:

```
FP 3 Busy CTRL 0: Command request has been submitted.
FP 3 Busy CTRL 0: Command passed.
```

The following excerpt from a procedure shows the command syntax used in this document:

1 Manually busy the CTRL on the inactive plane by typing

>BSY CTRL ctrl_no
and pressing the Enter key.

where

ctrl_no is the number of the CTRL (0 or 1)

Example of a MAP response:

```
FP 3 Busy CTRL 0: Command request has been submitted.
FP 3 Busy CTRL 0: Command passed.
```

Introduction

Publication application

The information contained in this publication applies to offices with Batch Change Supplement release 28 (BCS28) software. The editions for a given BCS release are contained in *Product Documentation Directory,* 297-8991-001.

Reason for reissue

This publication has been updated to include:

- a new report output by the PROCESS command
- a change to the component report to include the card code of the card on which the component resides
- the parameter CARDS which is added to the PROCESS command

Revision bars in the left margin indicate text that has changed or has been added since the last issue of this manual.

Software identification

Software applicable to a specific DMS-100 Family office is identified by a BCS release number and by Northern Telecom (NT) Product Engineering Codes (PEC). The significance of the BCS number and the PEC is described in *Provisioning Guide*, PLN-8991-104 (section 450/32) and in the Office Feature Record D-190.

A display of the BCS number and PEC for the NT feature packages available in a specific office can be obtained by entering the command string:

```
PATCHER;INFORM LIST;LEAVE
```

at a Maintenance and Administration Position (MAP).

Command descriptions

This publication uses a uniform system of notation to illustrate system commands and responses. It shows the order in which command elements appear, the punctuation, and the options. Where the conventions are not used, an explanation is given in the text.

Command descriptions consist of the following:

- Detailed Description

  Provides a detailed explanation of what the command does, system actions during the processing of the command, and other relevant information.

- Syntax

  Syntax is illustrated by means of a syntax box.

COMMAND	parameters

The syntax box is divided into two parts by a solid vertical line. The left part of the box contains the command name, and the right part contains the parameters associated with the command.

The following notational conventions are used to define the command syntax:

- Command names are not abbreviated.
- Constant parameters are written in uppercase letters. This indicates that they must be entered exactly as shown.
- Variable parameters are written in lowercase letters.
- Constant or variable parameters can be optional.

Parameter syntax is as follows: Optional parameters are enclosed in square brackets. A command entered without an optional parameter results in the default parameter value. In the following example the command may be followed by the optional parameter A, if required.

COMMAND	[A]

When one or more optional parameters may be selected out of a group of optional parameters, the parameters are arranged in a vertical list inside square brackets. In the following example the command may be followed by either of the optional parameters A, B, or C, if required.

```
COMMAND    ⎡ A ⎤
           ⎢ B ⎥
           ⎣ C ⎦
```

When one or more parameters must be entered, the group of parameters to select from, is shown in a vertical list. In the following example, the COMMAND must be followed by A, B, or C, or a combination of the parameters as specified in the detailed description of the command.

```
COMMAND      A
             B
             C
```

When a parameter selected from one group must be followed by a parameter from a second group, the sets of parameters are shown in vertical lists. In the following example, parameter A, B or C must be followed by parameter D, E or F.

```
COMMAND      A   D
             B   E
             C   F
```

A space is shown between sets of optional parameters to indicate those which must be followed by a parameter from a second vertical list located to the right of the first. In the following example, parameter A, B or C must be followed by parameter E or F. Parameter D must be followed by parameter G or H.

```
COMMAND      A   E
             B   F
             C

             D   G
                 H
```

Ellipses (...) are placed after parameters which may be input more than once.

COMMAND	(unit...)

When a command has so many parameters that they spill out of the right-hand side of a syntax box, the group of variable parameters is replaced by a substitute parameter.

Parameter List

A parameter list follows the syntax box in which the substitute parameter is defined. If necessary, a second parameter list is nested inside the first. The parameters which were substituted are defined. This is demonstrated as follows:

COMMAND	len1 len2

where

len1 consists of parameters:

```
frame1
unit1
drawer1
circuit1
```

where

frame1 is the starting frame number

unit1 is the starting unit number

drawer1 is the starting drawer number

circuit1 is the starting circuit number

len2 consists of parameters:

```
frame2
unit2
drawer2
circuit2.
```

where

frame2 is the end frame number

unit2 is the end unit number

drawer2 is the end drawer number

circuit2 is the end circuit number

WARNING
Impact of the command
An uncontrolled SWACT will occur. Do you wish to continue (Y or N)?

Responses

Responses that are displayed or printed on an output device are shown with an explanation for each response. Included in the response is the action the system takes and that the user should take as a result of a response.

Responses

```
INVALID TERMINAL IDENTIFIER
```

Meaning: Indicates that the node and terminal specified is not recognized by the system.

System Action: None.

User Action: Verify and re-enter the correct data.

Notes

Limitations, inconsistencies, and unusual situations that require a different procedure are described in usage notes.

Usage Note:

The SELECT command must be ON when using the INTERCEPT command.

Examples

Examples are often used to clarify a complicated command or to illustrate complex sequences of responses to a command.

Examples:

1 To monitor incoming messages enter:

>INTERCEPT IN MON
and pressing the Enter key.

```
PMIST SINGLE USER:
```

2 To disable both the incoming and the outgoing messages, enter:

>INTERCEPT BOTH OFF
and pressing the Enter key.

```
PMIST SINGLE USER:
```

References

References listed as ”prerequisite” are essential for an understanding of this publication. Informative references contain detailed information concerning other items mentioned in this publication, but are not essential. References are inserted at the appropriate places in the text.

Note: The documents listed may exist in more than one version. See *Product Documentation Directory*, 297-8991-001 to determine the release code of the version compatible with a specific release of software.

Prerequisite References

DOCUMENT NUMBER	TITLE
PLN-5001-001	*SuperNode Technical Specification*

Informative References

DOCUMENT NUMBER	TITLE
297-8991-001	*Product Documentation Directory*
297-YYYY-350	*Translations Guide*
297-1001-591	*Networks Maintenance Guide*
297-1001-595	*Trunks Maintenance Guide*
297-1001-520	*Maintenance System Man-Machine Interface Description*
297-1001-594	*Lines Maintenance Guide*

Test description

BERP test

The purpose of Bit Error Rate Performance (BERP) testing is to assess the bit error rate performance of the DMS switch and its hardware components.

BERP test is composed of many individual Bit Error Rate Tests (BERT). The craftsperson performs a BERT by connecting an Integrated Bit Error Rate Tester (IBERT) either to itself or to a specified endpoint such as a Data Line Card (DLC), and transmitting a known bit pattern. This bit pattern is reflected back to the IBERT and compared to what was sent. Any errors in the returned bit stream are recorded. The results of these individual BERTs comprise the result of the BERP test. The procedures for performing a BERP test are detailed in chapter 3 (BERP Testing Procedures).

Testing requirements

BERP testing is provided with feature package NTX881AB and requires either of the following Integrated Bit Error Rate Testers (IBERTs):

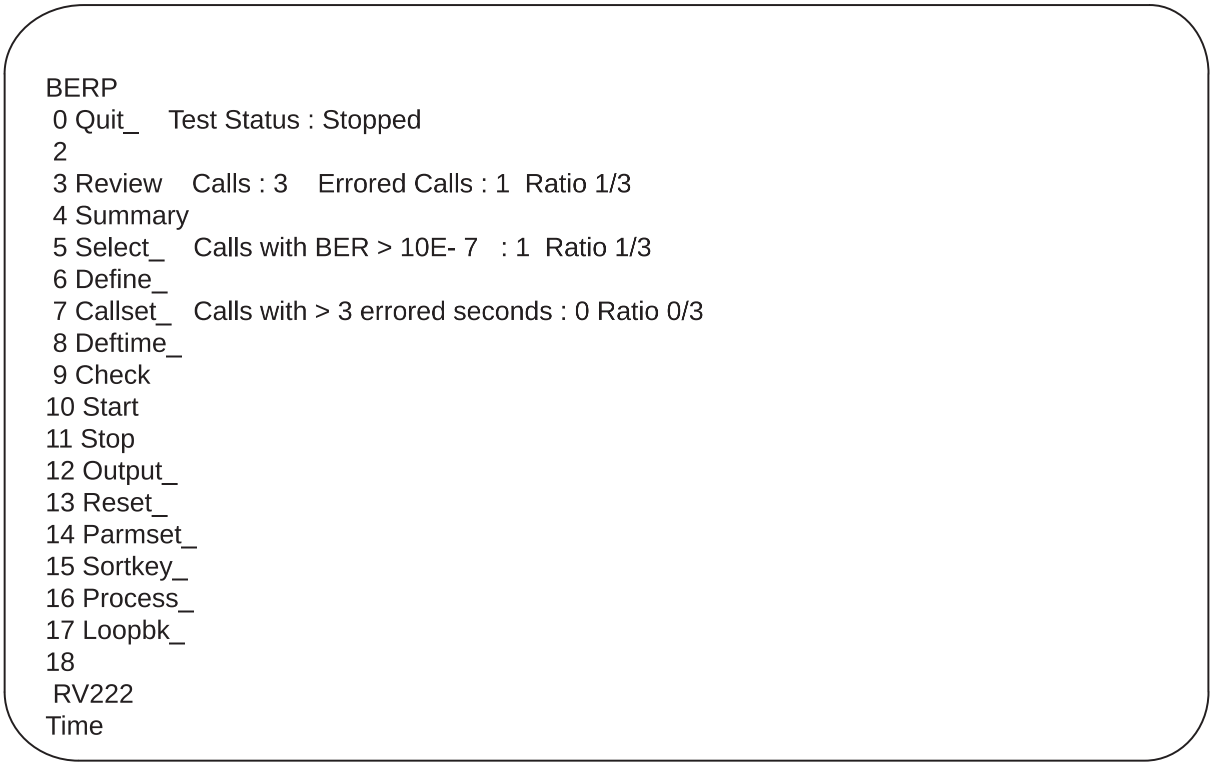
- NT4X23AA Digital Test Unit (DTU) located in a Maintenance Trunk Module (MTM)
- NT6X99AA IBERT Line Card (ILC) located in a Line Concentrating Module (LCM)

Bit Error Rate Performance tests are used for testing transmission paths through the network and provide the operating company with a tool for assessing the bit error performance of the DMS-100 Family switch.

BERP map level

BERP tests are available at the MTC level of the Maintenance and Administration Position (MAP). Commands at the BERP level are used to set up tests continuously or for a fixed duration. Figure 2-1 on page 2-2 shows how the BERP commands appear on the MAP.

Figure 2-1
BERP Level Display



```
BERP
 0 Quit_    Test Status : Stopped
 2
 3 Review    Calls : 3    Errored Calls : 1  Ratio 1/3
 4 Summary
 5 Select_    Calls with BER > 10E- 7   : 1  Ratio 1/3
 6 Define_
 7 Callset_   Calls with > 3 errored seconds : 0 Ratio 0/3
 8 Deftime_
 9 Check
10 Start
11 Stop
12 Output_
13 Reset_
14 Parmset_
15 Sortkey_
16 Process_
17 Loopbk_
18
 RV222
Time
```

The BERP level as shown in Figure 2-1, is updated in real time with the following key statistics:

- the test status
- the number of calls made
- the number of errored calls
- the number of calls with a Bit Error Ratio (BER) worse than the value specified by the user
- the number of calls with more errored seconds than the value specified by the user

Testing configurations

The IBERT used for BERP testing can be either an NT6X99AA IBERT Line Card in a Line Concentrating Module, or an NT4X23AA Digital Test Unit (DTU) in a Maintenance Trunk Module. The following are the various configurations for BERP testing.

DLC testing

Typical BERP test configurations for data line cards (DLC) are shown in Figure 2-2 through Figure 2-5 on page 2-6 . To perform a BERP test, DLCs are defined in a pool for testing. There must be at least one DLC for each IBERT to be used. If there are more DLCs than IBERTs, the IBERTs are assigned to the DLCs on a rotating basis.

Figure 2-2
Typical IBERT to DLC configuration with an ILC in an LCM

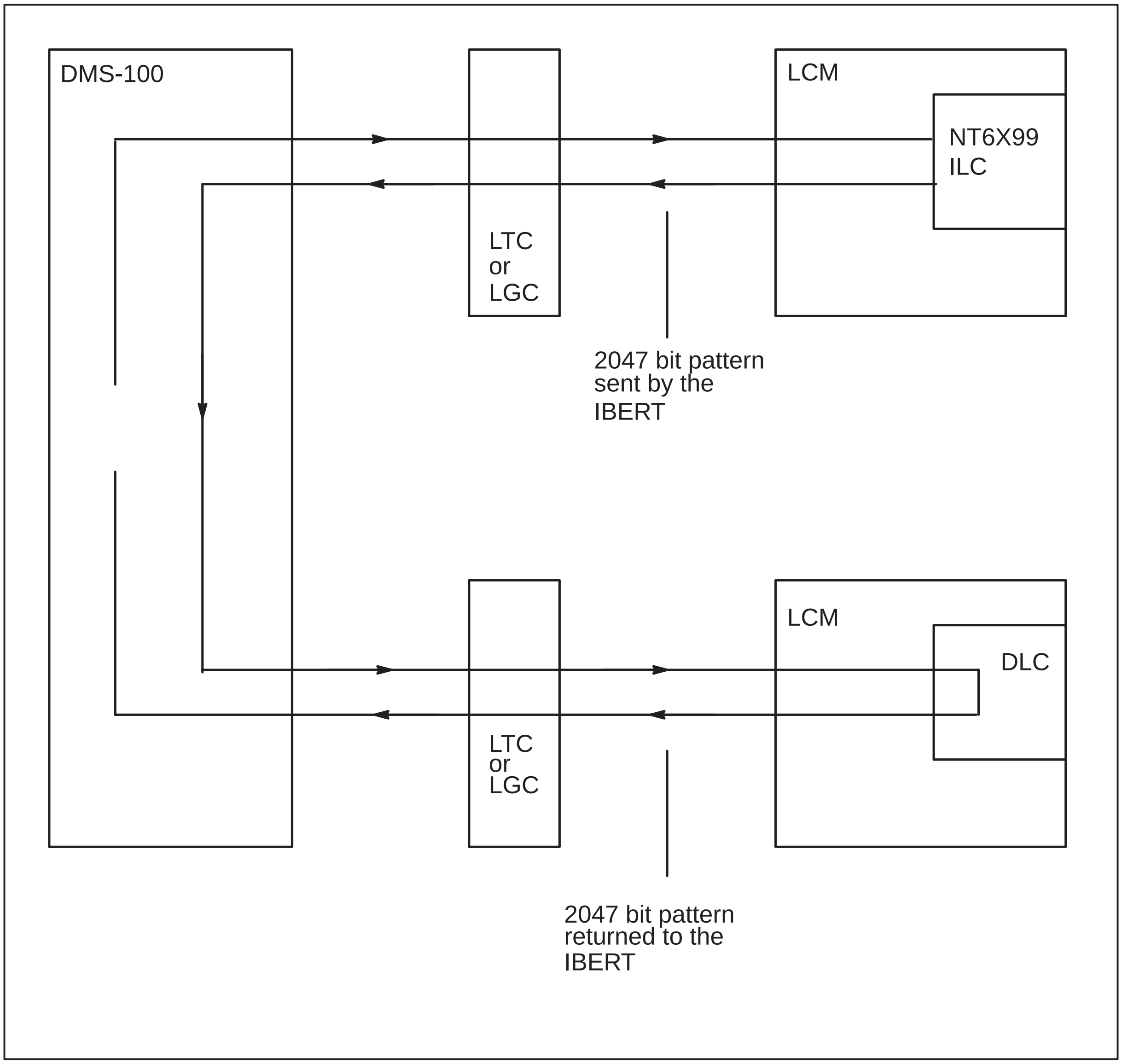


Figure 2-3
Typical IBERT loopback with an ILC in an LCM

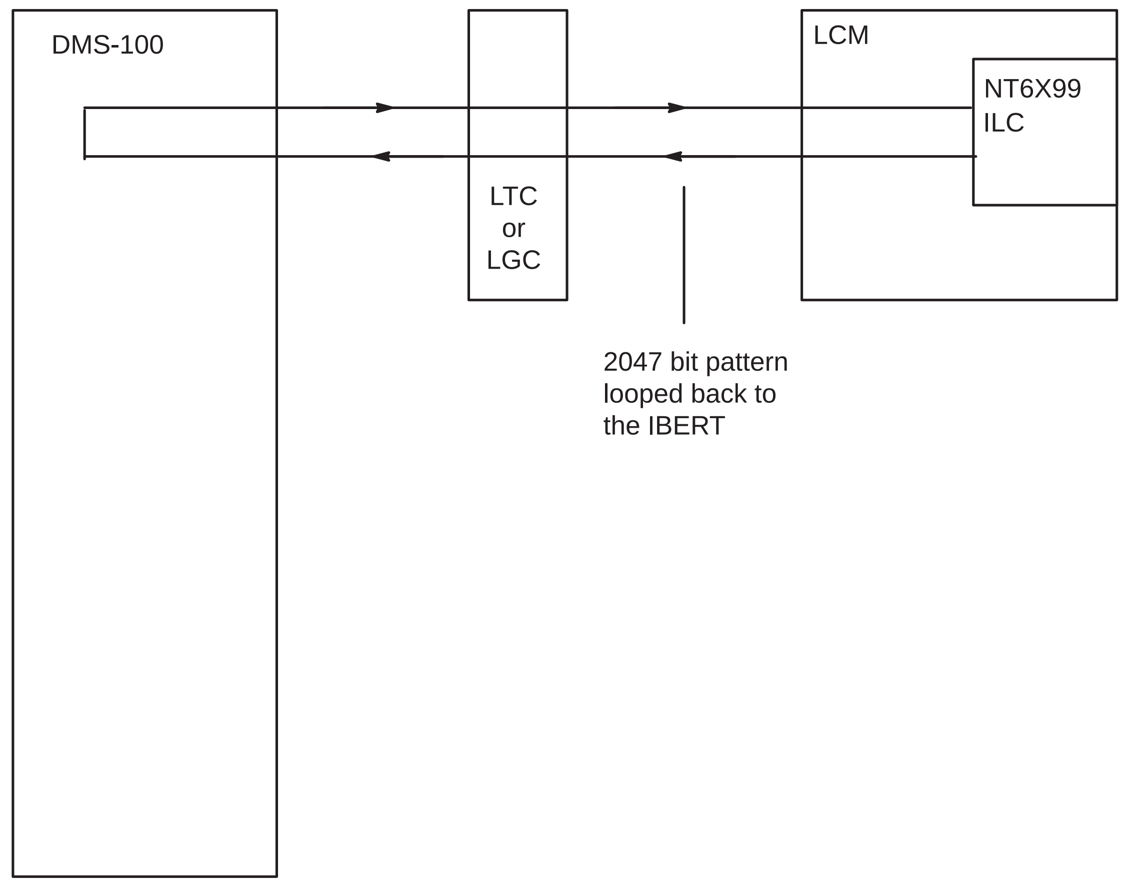


Figure 2-4
Typical IBERT to DLC configuration with a DTU in an MTM

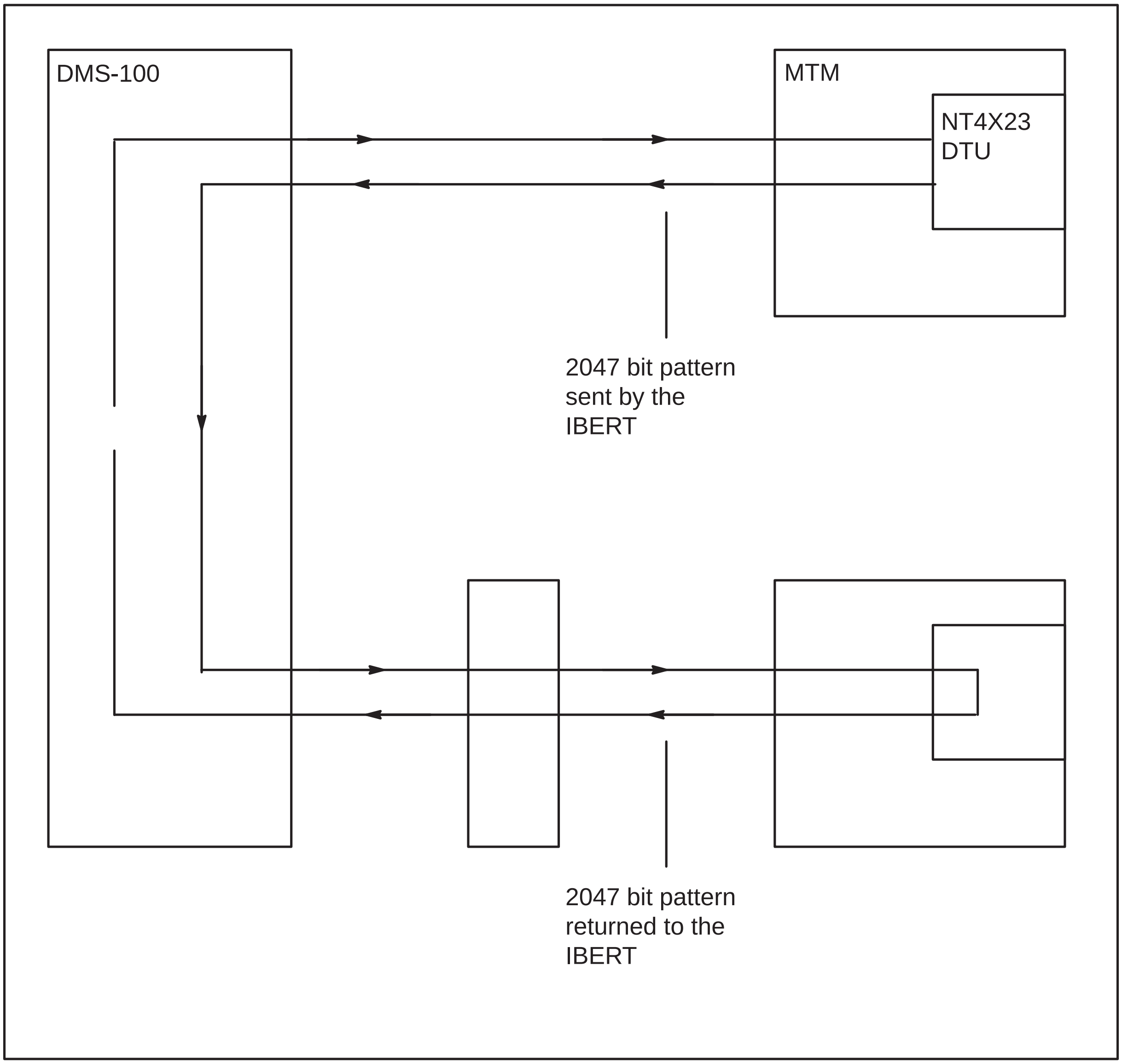
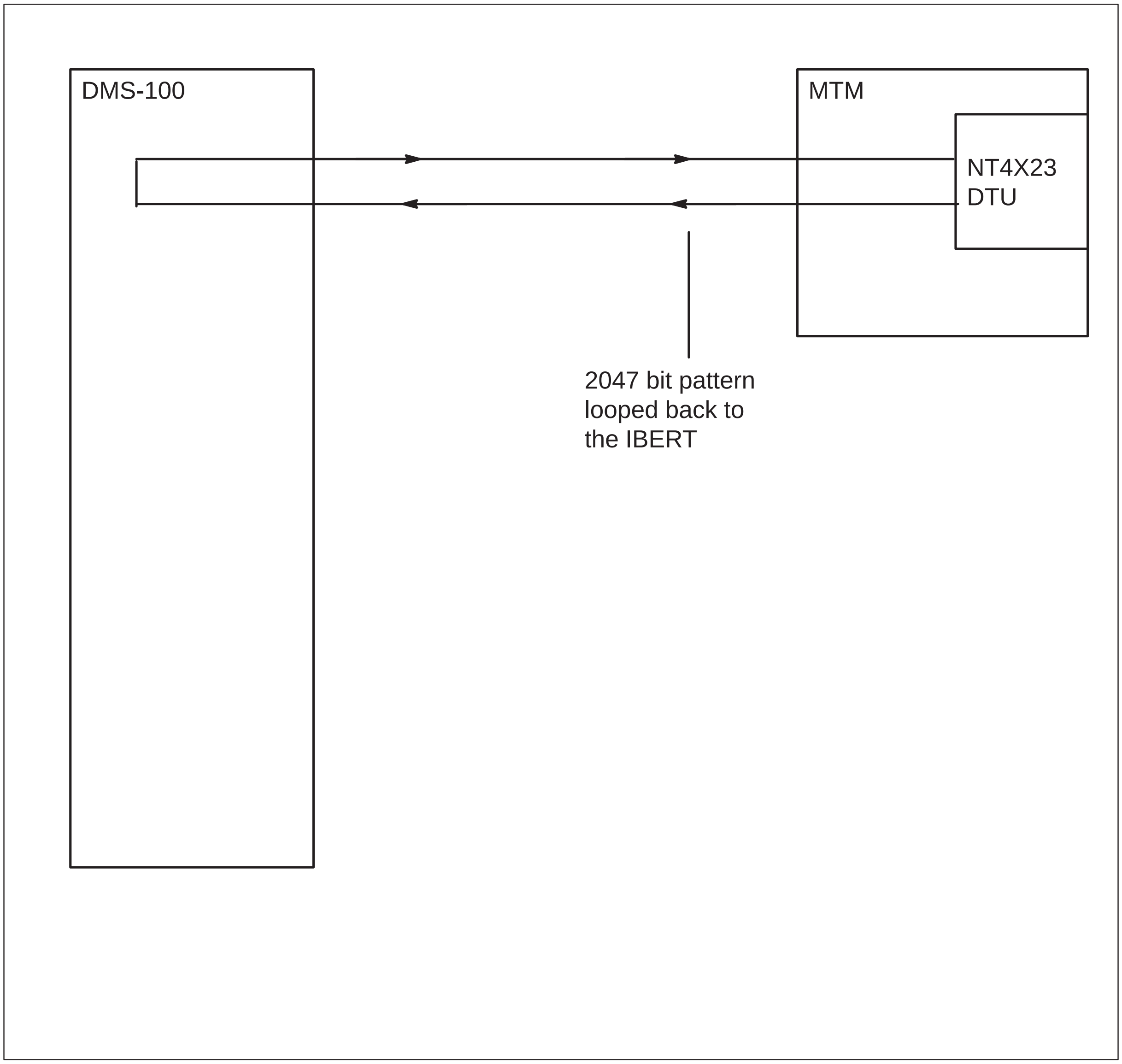


Figure 2-5
Typical IBERT loopback with a DTU in an MTM



Trunk/link testing

To perform a BERP test on a trunk or a link it is necessary to have a loopback at some point on the transmission path. The loopback can be specified at one of the following points:

- NT6X48 DS30 interface card (any channel)
- NT6X50 DS1 interface card (full loopback all channels)

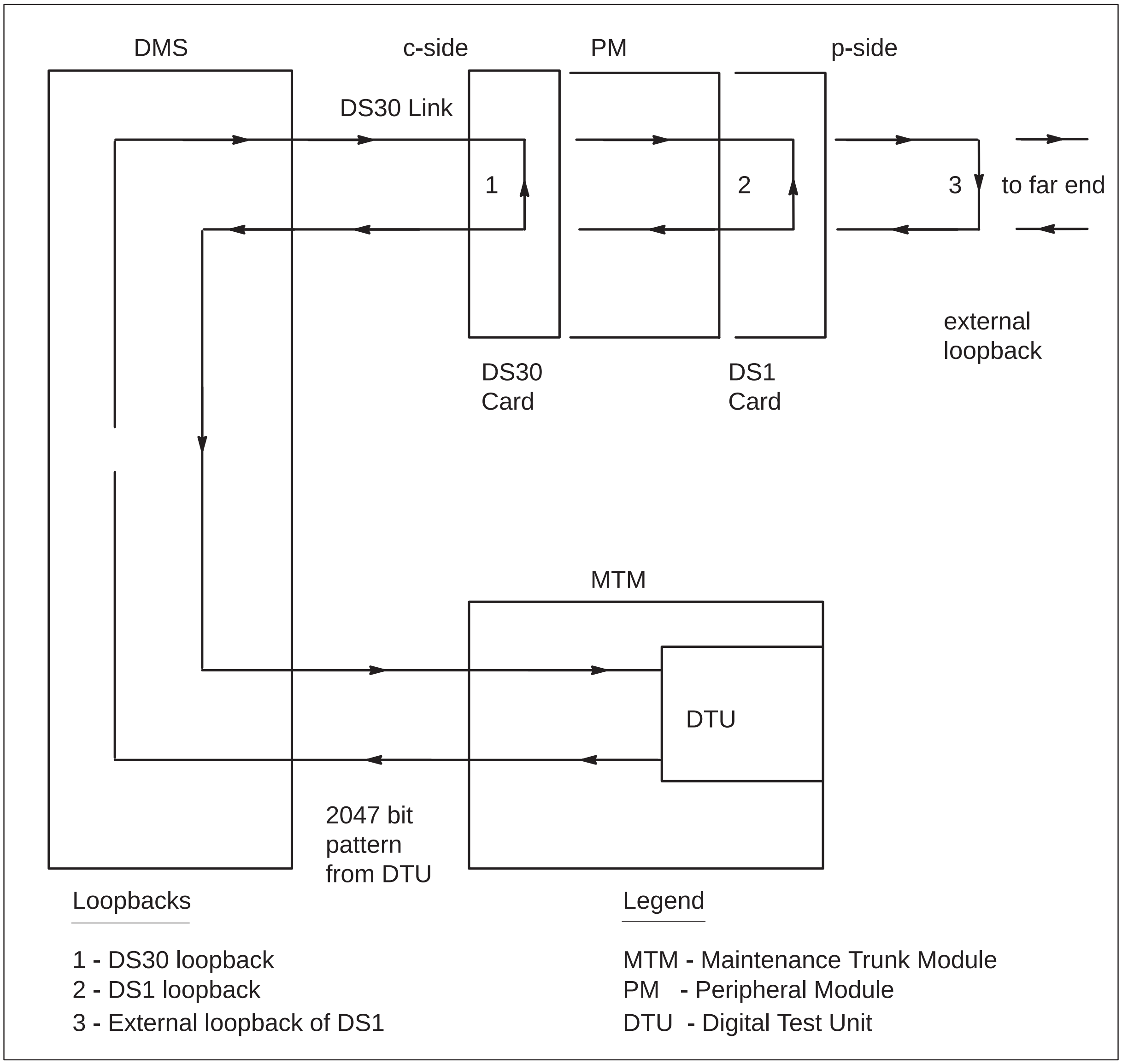
- external loopback

These loopback points are shown in Figure 2-6 on page 2-7.

Note: This manual defines a trunk as being one channel on a link.

BERP testing applies only to trunks with a SIGNALING DATA SELECTOR of STD, STDTL, R1 or R1N5. A TRUNK GROUP TYPE of MTX or RONI cannot be tested.

Figure 2-6
Typical BERP loopback points for trunks/links



Loopbacks

1 - DS30 loopback
2 - DS1 loopback
3 - External loopback of DS1

Legend

MTM - Maintenance Trunk Module
PM - Peripheral Module
DTU - Digital Test Unit

The DS30 loopback is at the c-side of the peripheral module (PM) and can be defined for a specific channel (trunk) without affecting other channels on the DS30 link. When a trunk is tested on a Remote Cluster Controller (RCC), the actual loopback is on the DS30 card on the LTC.

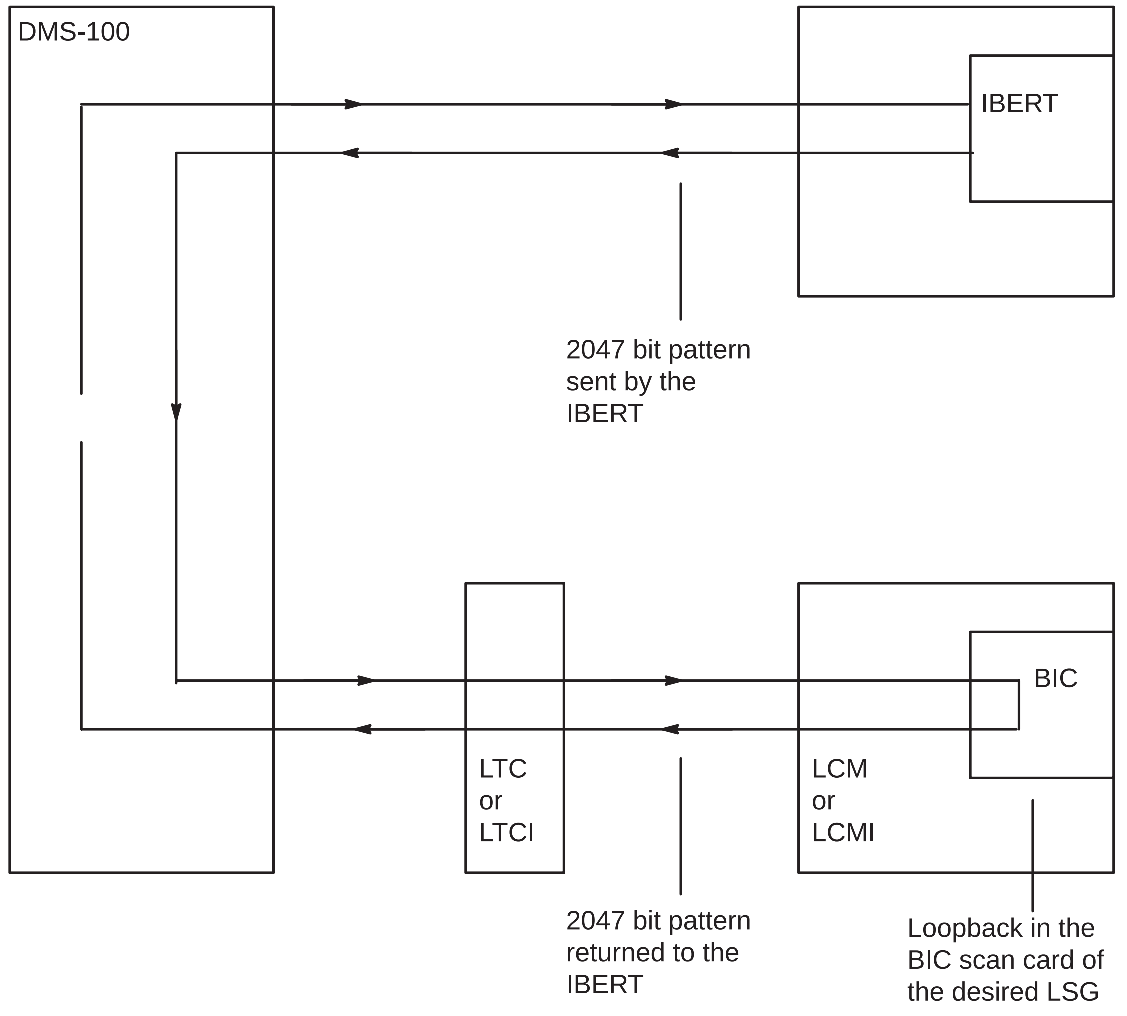
The DS1 loopback is at the p-side of the PM and all 24 channels on the DS1 link are looped back. When a DS1 loopback is used, the the DS1 must be removed from service. If the DS1 loopback has been set at the CARRIER level, the LOOPBK DS1 command cannot be used. For more information on CARRIER level testing see Carrier Level Tests on page 2-20.

Trunks can be defined for testing in the same pool as DLCs. Trunks must be properly datafilled. Links cannot be defined in the same pool as trunks and DLCs. The pool of trunks and DLCs must be cleared before defining links for testing. Similarly, if a pool of links is defined for testing, the links must be cleared before trunks and DLCs are defined.

LSG looparound

An IBERT is connected to a line in the LCM or LCMI in order to establish a connection to a Line Subgroup (LSG). Once this connection is established, its P-side portion is replaced by a new connection to the scan card of the desired LSG. The scan card for an LSG is located on the Bus Interface Card (BIC) of the corresponding drawer of the LCM or LCMI. This configuration is shown in Figure 2-7

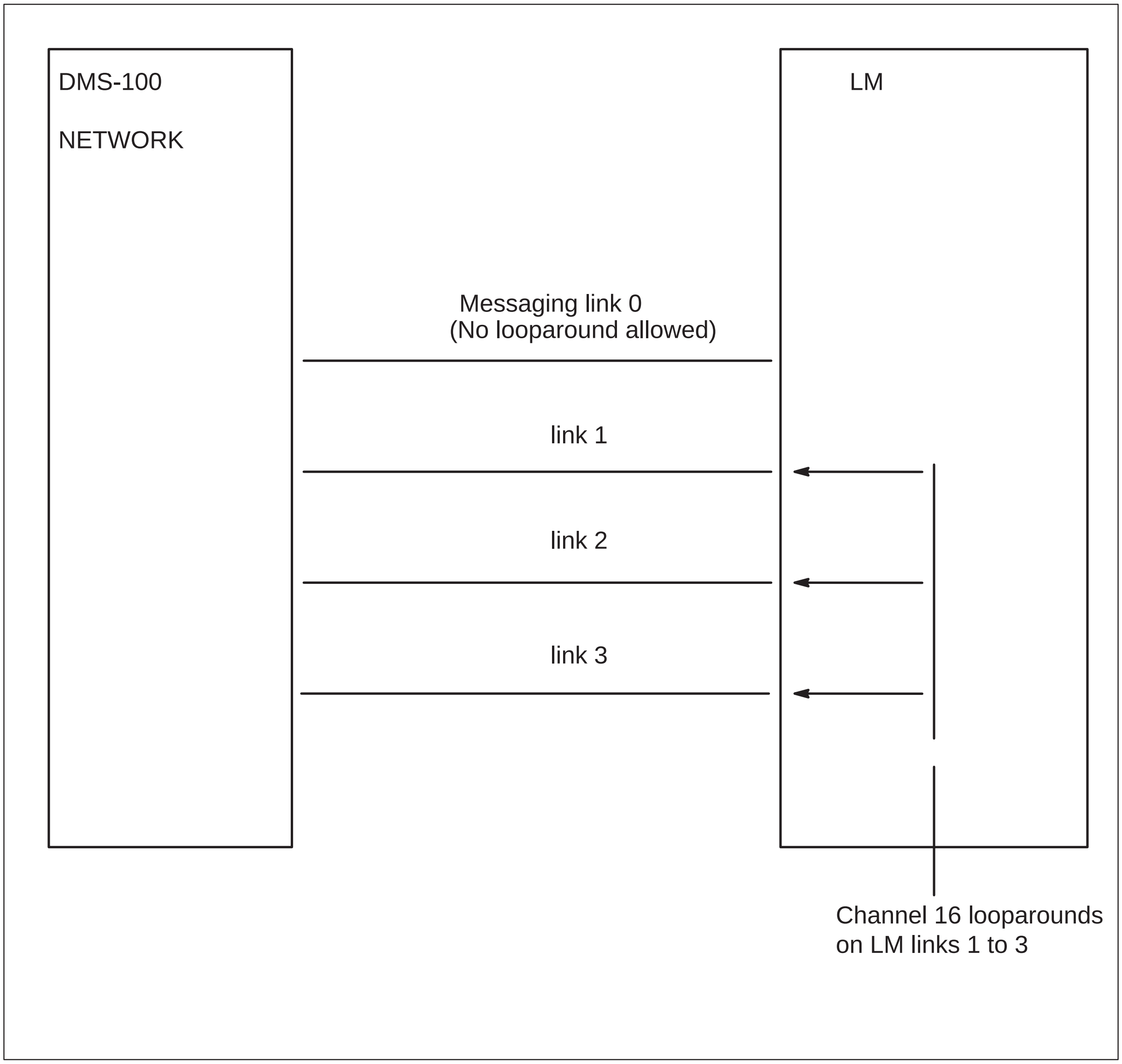
Figure 2-7
Typical IBERT to LSG configuration


DMS-100
IBERT
2047 bit pattern sent by the IBERT
BIC
LTC or LTCI
LCM or LCMI
2047 bit pattern returned to the IBERT
Loopback in the BIC scan card of the desired LSG

LM looparound

BERP testing for the LM links is provided by the channel 16 looparounds on the links which connect the LM to the DMS network. Links 1 to 3 are used for the looparounds which are made at the link interface of the LM. Link 0, which is used as messaging link between the DMS network and the LM, cannot be used for looparound. This configuration is shown in Figure 2-8 on page 2-10

Figure 2-8
Typical LM to Network configuration



IBERT resource management

Resource management for IBERTs involves the ability to reserve an IBERT for use by a specific application such as the Line Test Position (LTP), or to share the same IBERT among different applications such as the LTP and BERP. Under certain conditions, a priority application like the LTP is allowed to seize an IBERT from a lower priority application such as BERP. To relate a given IBERT to a given application (LTP, BERP etc), IBERT resource management is provided through table control. The following tables are required to be datafilled:

- Facility Maintenance Resource Inventory (FMRESINV) table 2-2
- Facility Maintenance Resource Users (FMRESUSE) table 2-1

All IBERTs, both ILCs and DTUs, must be datafilled in table 2-2 for any of them to be used as an IBERT by any application. In addition, the IBERTs must still be datafilled in their other associated tables: LNINV for ILCs, and TRKMEM for DTUs. The following is a sample datafill for the FMRESUSE and FMRESINV tables:

Table 2-1
FMRESUSE

```
BERP C (0) (1) $
LTP ALL
```

Table 2-2
FMRESINV

```
IBERT 0 L  0 0 0 0 C (0) $
IBERT 1 L  0 0 0 1 ALL
IBERT 2 G DTU 0       NONE
IBERT 3 L  0 0 1 0 C (1) (4) $
```

This sample shows that:

- BERP is allowed to use IBERT 0, IBERT 1 and IBERT 3
- LTP is allowed to use all IBERTs except IBERT 2

The use of a particular IBERT by a particular application is determined by their respective CLASS assignment in FMRESINV and FMRESUSE tables. The CLASS can be set as one of the following:

- ALL means all classes (0 to 15) are assigned to an IBERT and it can be used by any application
- NONE means no class has been assigned and the given IBERT cannot be used by any application
- a set of numbers from 0 to 15 means the IBERTs that have CLASS numbers matching those of the applications can be used by those applications

For more information on data tables, refer to *Translations Guide* 297-YYYY-350

Impact on BERP

The resource management of IBERTs has some impact on BERP testing when BERP has to share its IBERTs with other applications such as the LTP. Due to space constraints, for example, lack of line card space in the LCMs, or other reasons, it is not practical to reserve the required number of IBERTs for use by every type of application. Sharing of IBERTs by some applications is, therefore, more desirable. When BERP shares an IBERT with the LTP, the LTP can claim the IBERT being used by BERP, provided the LTP was unable to obtain an IBERT from those reserved for LTP use. The LTP, when claiming an IBERT from BERP, interrupts BERP. The impact of this interruption on BERP depends on the following two factors:

- number of calls that BERP is making at the time of the interrupt
- number and frequency of interrupts

If the number of calls is high, and the number and frequency of interrupts is low, the impact on BERP is minimal. Otherwise, it is advisable to datafill more IBERTS exclusively for the application such as the LTP.

The interrupted call is aborted and is not included in the BERP statistics. However, the number of interrupted calls is recorded in the same manner as the number of call failures.

IBERT status

QBERT is a CI command which can be used to check the status of any IBERT. At the CI level, enter the following command:

```
> QBERT IBERT nbr
```

where nbr is the IBERT number from 0 to 127

Along with the other information, it provides the status information of the IBERT. The IBERT status can be one of the following:

- BERT OK
- BERT SUSPECT
- BERT FAULTY

BERT OK means that the IBERT is good for use in a BERT test. BERT SUSPECT means at least one failure to start a BERT test has occurred while using this IBERT. BERT FAULTY means that the IBERT has failed a diagnostic.

When an IBERT fails to start a BERT test, its status is changed to BERT SUSPECT and a BERT100 log is generated to record the fault. The IBERT is then put on the appropriate diagnostic shower queue: Line shower queue for ILCs and the trunk shower queue for DTUs. The system runs a diagnostic on the IBERT. The results of the diagnostics are indicated by the following logs:

- LINE100 (pass) for an ILC
- LINE101 (fail) for an ILC
- TRK107 (pass) for a DTU
- TRK106 (fail) for a DTU

If an IBERT passes the diagnostic, the suspect status of the IBERT is changed back to BERT OK; if, on the other hand, it fails diagnostic, the status is changed to BERT FAULTY.

Test statistics

The major difference between measuring bit error performance with BERP and using external test methods, is that the performance summary generated by BERP aides in the isolation of the sources of errors occurring along the transmission path. An external test set can determine only that errors occurred, while BERP tests provide a detailed summary of all the paths involved.

It is up to the operating company to determine an acceptable level of bit error rate performance for the switch. A bit error rate is the number of bits in error in a given transmission stream. The bit error rate is expressed as an absolute ratio such as

```
1.0 x 10E-9
```

This Bit Error Ratio (BER) implies that out of one billion bits sent, one was in error. Since it is not practical to set up test calls of a billion bits, the number of error free seconds is measured in a test call of at least 10 minutes duration. The percentage of error free seconds in a test call is then used to extrapolate the true performance of the data transmission. The number of error free seconds plus the number of error free calls made is used to generate a statistical confidence level of the bit error rate performance of the switch.

It is up to each operating company to take the statistics generated and determine whether the switch is performing to their standards.

The statistics gathered during a BERP test are a summation of the results of each of the individual Bit Error Rate Tests (BERT). A call is considered as having been made if the IBERT is successfully connected to an end point and a test is started. Note that this does not imply that any data is passed.

The first stage of a call of this type is to achieve synchronization (sync). This is done by an internal protocol which allows both ends to identify their operating parameters and to decide which set of parameters to use in the test. The IBERT always adapts to the operating parameters of the other end. If for some reason this protocol fails, data is not passed. This is represented as a ”no sync” call. A no sync call is considered to be an errored call since it represents a condition so severely errored that no data could be passed. An errored call is any call in which either no sync is found or the number of bit errors is greater than zero.

When the IBERT ”syncs up” with the end point, it begins to transmit its 2047 bit pattern and monitor the returned pattern looking for any errors. The transmission speed used is determined by whether or not the IBERT or its end point are located at the host. If both the IBERT and its end point are at the host, the maximum speed of 64 kbps is used. If either the IBERT or its end point is located on a remote site, then the 56-kbps speed is used. This is because the maximum transmission rate over the links connecting a remote site to the host is 56 kbps. All trunk BERP testing is performed at 56 kbps.

After the call has been active for the user-specified length of time, the IBERT is queried for the results of the test. Since the call was completed, the CALLS MADE statistic is always updated. If no errors occurred, then the ERROR FREE CALLS statistic is incremented. If errors occurred, the ERRORED CALLS statistic is updated. The bit error ratio (BER) of the call is then calculated and compared to the value specified by the user. If the BER is greater than the user-specified value, the statistic for calls worse than the specified ratio is incremented. If the number of errored seconds in the call is greater than the value specified by the user, the count for the NUMBER OF ERRORED CALLS is incremented. If any errors are found, the path involved in that call is traced and a report on the errored call is placed on an output file. A trace is not performed if an output file is not specified.

When the LTP that shares an IBERT with BERP seizes that IBERT from BERP, it interrupts the BERP testing. Every time an interruption occurs, BERP keeps track of it and includes it in the summary display at the end of the BERP test.

Any failure to seize the IBERT (put it into an MB state for use during the test) is reflected by the seize failure count. If Data Line Cards (DLC) are being used in the test, any failure to seize a DLC is also reflected in this count. Any failure to connect the IBERT to its end point is reflected in the call set-up failure count. Neither seize failure nor call set up failure is considered a call. For this reason, neither is considered as being errored.

Additional test tools

If performance errors are detected at the BERP level, other tests are available to further diagnose the problem. The following paragraphs provide an overview of these tests:

- Network Path Diagnostics (NET PATH)
- Integrity Check Traffic Simulator (ICTS)
- XPM Bit Error Rate Testing (XBERT)
- Carrier Level Testing

Additional information on procedures and command syntax for these test tools can be found in the Northern Telecom publications referenced with each description.

NET PATH

The Network Path Diagnostics (NET PATH) test tool is available in Feature Package NTX885. NET PATH is documented in *Networks Maintenance Guide*, 297-1001-591

Offices with this feature can access the NET PATH level from the NET level by entering the command PATH or menu item 14.

The NET PATH test tool assists in transmission path maintenance by:

- identifying faulty components that cause integrity failures
- confirming that suspect components are faulty before they are replaced
- verifying whether or not replacing the components has corrected the fault(s)

The NET PATH feature supports the following network types:

- NT5X13
- NT7X27

- NT7X40
- NT8X11

Network types NT5X13, NT7X27, and NT7X40, require network firmware release 3 or greater. Due to hardware restrictions, the NET PATH feature cannot be implemented on Network type NT0X48.

NET PATH tests are performed by inserting a test pattern at a point along a transmission path, and extracting it at another point further along the path. (Both the insertion and extraction points can be selected by the user.) An extracted pattern that differs from the inserted pattern indicates the presence of faulty hardware in the path segment under test.

To isolate the component, a series of NET PATH tests with different insertion and extraction points can be run on the path. Subsequent tests using the original insertion and extraction points will verify that replacing the component has corrected the fault.

Tests are automatically scheduled on the required resources in the order in which they are submitted. However, tests run only when all of the required resources become available. While a test is in progress, test results and test data are constantly updated and echoed in the status display area of the NET PATH level. In addition, the results of one or all of the submitted tests can be queried and displayed at any time.

NET PATH tests can be run on any PM directly connected to a Network, with the exception of Line Modules (LMs). NET PATH tests require trunks in the INB state, and may be aborted if the trunk changes state while the tests are running.

A record containing input and output data and test control information is created for each path to be tested. Twenty records are available, all of which may be submitted for testing at the same time. However, if more than one test is submitted on a record, they form a queue and run sequentially.

To change the information in a record, it must first be posted. (When a record is posted, it is displayed at the MAP and can be acted on by the NET PATH commands.) To avoid the possibility of two users altering the same record at the same time, only the first user to post a record can change the data or terminate a test running on the record. Another user may view the posted record, but cannot take control of it until the first user has cleared the display.

Tests run unattended; therefore, the user may submit the tests and leave the NET PATH level. When the tests are completed, the resources are freed but test results remain in the records until they are cleared by the user.

Commands to define, monitor, and display the progress of the tests are entered at the NET PATH level of the MAP.

ICTS

The Integrity Check Traffic Simulator (ICTS) is available in Feature Package NTX885. ICTS is documented in *Networks Maintenance Guide*, 297-1001-591.

ICTS assists in speech path maintenance by identifying paths that have faults caused by marginal hardware components. Using call processing resources, ICTS simulates traffic by setting up connections on call paths between Networks and the PMs associated with them. The PMs involved in the connections continuously check integrity on the same plane after integrity failures are reported. Since integrity counters are incremented at each check, faulty paths are identified by the high integrity counts registered against them.

Note: ICTS tests use call processing resources, and can compete with call processing for Network paths. To avoid such competition, run ICTS tests only when traffic is low.

ICTS can be used simultaneously with traffic or Call Through Simulator (CTS) load boxes for continuous monitoring of Network links. All the commands available with ICTS are non-menu commands and can be accessed by entering the command ICTS at any level of the CI, MAPCI or MAP.

ICTS connections are set up on links specified by the user. If the specified links meet ICTS requirements, they are configured, that is, marked as available for ICTS connections. ICTS scans the set of configured links until two available channels are found. Network to PM connections are then set up, and the PMs at the pathends begin integrity checking. When an integrity failure occurs, the PM reporting the failure sends a message to the CC and switches Network planes. The CC verifies that the connection belongs to ICTS, increases an integrity count against the PM and Network hardware which reported the failure, and generates Log NET100 or 102.

To ensure accurate integrity counts against faulty hardware, ICTS forces the PMs to continuously check integrity on the Network plane on which the integrity failure was reported. For example, if an integrity failure occurs on plane-0, the PM reports the failure and switches to plane-1. To obtain repeated failures on plane-0 (and, therefore, accurate integrity counts against the faulty hardware in the path), the PM switches back to plane-0 and continues checking on the original plane.

If the quantity of integrity failures detected exceeds the integrity threshold, ICTS turns off integrity checking by the PMs, clears the ICTS connection, and generates an ICTS101 Log. The paths on which the failures are detected are recorded in the path fault buffer and the NET INTEG buffer for further testing.

Note: Connections with high integrity counts are stored in the path fault buffer. When testing paths identified as faulty by the ICTS tool, check the path fault buffer first.

Paths that are set up by the NET PATH feature can be tested by ICTS for fault identification. ICTS first ensures that the connections are valid, and that the PMs involved are supported by ICTS, then starts integrity checking. Connections are rejected if they do not meet ICTS specifications.

XBERT

The XPM Bit Error Rate Test (XBERT) is provided with feature package NTX885. XBERT is documented in *Peripheral Modules Maintenance Guide*, 297-1001-592. XBERT detects bit errors in the transmission of high-speed data in LCM and RLCM cards, and the following PMs:

- DTC
- IAC
- LGC/ ILGC
- LTC
- MSB7
- RCC.

Note: To use XBERT, each of the PMs must be equipped with an NT6X69AB message card or an NT6X69AA message card with an NT6X79 tone card.

XBERT is capable of six separate tests which handle different hardware components in the PM transmission paths. The test names and their corresponding cards are:

- XPMINT (internal) NT6X41 NT6X42 NT6X44 NT6X48 NT6X69
- XPMPSL (p-side loop) NT6X44 NT6X48 NT6X69
- XPMDCC (digroup conrol cards) NT6X44 NT6X48 NT6X69 NT6X52
- 0 XPMBIC (bus interface cards) NT6X44 NT6X48 NT6X69 NT6X52 NT6X54
- 0 XPMHLP (hard loop) NT6X44 NT6X69 NT6X50/NT6X27 and associated carrier.

- ISOLATE automatically runs tests to isolate a fault to a particular set of cards. The number of cards in its card list can vary between one and three cards, depending on the individual test results.

P-side ports or LCM Bus Interface Cards (BIC) can be tested sequentially by one manual request.

In each of the XBERT tests, if the PM P-side port being tested is a DS-1 port instead of a DS-30A port, then the NT6X50 card is tested instead of the NT6X48 card. For tests XPMDCC and XPMBIC, if the node on the P-side of the PM is an RLCM instead of an LCM, then the NT6X73 Link Control Card and NT6X50 cards on the RLCM Host Interface Shelf are also tested.

For accurate fault detection tests, each of the XBERT tests must be run on an active in-service PM unit. For test XPMDCC and XPMBIC, at least one unit of the LCM or RLCM must be in service.

Note: XBERT should not be used as a tool for providing accurate bit error rate assessments. It does not use the CCITT standard test patterns in its test procedure. Instead, it uses PM tone PCM to provide the 64-kbps test-bit stream.

All of the tests function in the same manner. Each of the tests checks:

- presence of specific hardware
- channel and data connections
- concurrency of tests.

All of the tests initially check for the presence of the NT6X44 timeswitch and the NT6X69 message cards. If these cards are not accessible, then XBERT displays a response and the remainder of the test is aborted. If the cards are accessible, XBERT then checks for the presence of the PM P-side interface card (NT6X48 or NT6X50) that controls the port that is to be tested manually.

Connections

If all of the hardware presence tests pass, then the individual XBERT tests are invoked to set up the channel connections for their individual test paths. Once the test path is set up, XBERT sends data through the looped test path and verifies it as it returns through the loop. Verification continues for up to nine hours, or until the manual testing.

XBERT can be run simultaneously on all of the valid PM types in an office. However, XBERT cannot test more than one test path at a time in any single PM unit. Although the P-side ports can be manually requested for testing, a specific channel on that port cannot be tested unless test XPMHLP is run, for which a channel must be specified.

XBERT tests run on in-service units so that there is no degradation of call processing while the tests are running.

At the end of a test, XBERT releases the test path connections and displays the bit error statistics based on the completed test run. These statistics can also be displayed at any time during the test run.

Carrier level tests

A DS1 loopback can be operated from the CARRIER level of the MAP for NT6X50 cards on PMs. For NT6X50AB DS1 cards it is possible to operate a DS1 loopback towards the the far-end (remote loopback).

A degree of fault isolation can be obtained between two offices by using the carrier loopback in conjunction with BERP tests as outlined in the following example:

1 In the near-end office post the carrier and make the carrier manbusy.

2 Set the local (L) DS1 loopback towards the near-end NT6X50AA/AB card by entering the following command string:

```
> LOOP n L
```

(where n is one of five currently posted carriers)

3 Test the link from the BERP level with the loopback option set to NONE.

4 Release the local (L) loopback by entering the following command string:

```
> LOOP n C
```

(where C clears the loopback at the NT6X50 card)

5 In the far-end office operate the remote (R) loopback towards the far-end NT6X50AB DS1 card by entering the following command sting:

```
0 > LOOP n R
```

6 To isolate the faults between offices, repeat BERP testing from the near-end office

Since looping a carrier causes alarms to come up, carrier testing should be coordinated between the offices involved.

For more information on the CARRIER level, refer to *Trunks Maintenance Guide*, 297-1001-595.

BERP testing procedures

STATUS CONDITIONS

For a better understanding of the bit error rate performance (BERP) tests, read the following paragraphs about STATUS CONDITIONS before performing a BERP test.

Each phase of a bit error rate performance test has a status to indicate the state of the BERP test. The following are all the values that the test status conditions may be:

- UNCHECKED
- CHECKED
- WAITING
- RUNNING
- STOPPING
- STOPPED

The UNCHECKED status means that data concerning the test set up is still being input, but has not yet been checked for consistency. Since the commands which set up the test data can be issued in any order, only basic checking of the data is performed at input. For instance, when Data Line Cards (DLCs), trunks or links are being input, the only check performed is for proper datafill. It is only when the START or CHECK commands are issued that further checking is done to ensure that the number of endpoints is equal to or greater than the number of IBERTs to be used.

A status of CHECKED indicates that all the user-supplied data for a test set-up has been verified for consistency. The CHECK command performs checks on the inter-dependencies of some of the data. Since these inter-dependencies cannot be checked when the input is supplied (because the commands can be issued in any order), these checks can only be performed before a test is started. The data may be checked either by issuing the CHECK command or by issuing the START command. The same data consistency checks are performed by both commands. Also, all data is checked during this process and an error or warning is generated for each check that fails. This is to overcome the problem of having to check the data each time one piece of information is corrected, only to discover that another piece of information has to be corrected. Note that when a test has reached the CHECKED status, changing any of the test set-up data resets the test status to UNCHECKED.

A status of WAITING indicates that a test is scheduled to run at a later time. This occurs when a START command is issued and the test set-up data includes a specific start time. A test in this state is similar to a RUNNING test and hence must be STOPPED in order to change any of the test set-up data. If any command is issued to change the test data, an error message is generated, stating that the test must be stopped first. To do this, the STOP command must be issued. The test status changes to STOPPED at this point and the data can be modified.

A status of RUNNING indicates an active test. When a specific start time is not supplied in the test set-up data, a test begins to run as soon as a START command is issued and all the data is verified. If a specific start time is supplied, a test becomes RUNNING when that time arrives.

When the test status is RUNNING, one of the following is displayed:

- calls setup
- time at which the current group of calls is to be taken down
- calls to be taken down
- time at which the next group of calls is to be setup.

A test status of STOPPING is a transient state. It may not appear on the MAP display due to the nature of the activity. This status is present once the stop criteria for the test is met, and persists until all test calls are fully stopped. In the extreme case of one test call being active when the stop criteria is met, the time taken for the test status to move from STOPPING to STOPPED is minimal. The MAP only updates every fifteen seconds and therefore may not display the interim status of STOPPING.

A status of STOPPED indicates that the test has completed. This may be due to the stop criteria being met, or a problem encountered writing to the output file. The exact reason can be determined from the SUMMARY command. Once the test has stopped, new test set up data may be entered for another test. When new data is entered, all counts associated with the last test are cleared, including the stop reason, and the test status goes back to UNCHECKED.

When any test parameter is altered, the test status is set to UNCHECKED. Changing any of the test parameters is disallowed if the test status is one of RUNNING, WAITING or STOPPING. A test can only be elevated to the CHECKED state by the commands CHECK and START, if they pass. The only command that can elevate the test status further is the START command. START sets the status to RUNNING or WAITING depending on whether a start time was issued. In order to set up a new test when the status is WAITING or RUNNING, the STOP command must be issued. This sets the test status back to STOPPED and new data may be entered, or existing data changed.

Performing a BERP test

The following paragraphs show how to set-up and run BERP tests using DLCs, trunks, links, or LSGs . The sequence of instructions does not have to be followed exactly as it appears in this section. The sequence that is followed depends on the amount of data that must be added or changed since the last BERP tests were performed. Each procedure has examples on the typical use of the command. Details on command syntax and system responses are in the section ‘BERP command and reponses’ in this document.

Entering the BERP level

To access the Bit Error Rate Performance (BERP) level of the Maintenance and Administration Position (MAP), enter the following command string after logging-on to the MAP:

```
MAPCI;MTC;BERP
```

The BERP level can also be accessed from the MTC level of the MAP by selecting menu item 5 or BERP.

Setting performance parameters

To set the BER exponent parameter and the errored seconds parameter, use the PARMSET command. Both of these are optional parameters and can be specified individually or together. The default values are a BER (Bit Error Ratio) exponent of 7, and 3 errored seconds. These values are displayed on the MAP. This means that any call with a Bit Error Ratio greater than 10E-7 or any call with more than three errored seconds is recorded as not meeting the operating standards. Any call with a BER less than 10E-7 or with three or less errored seconds is still recorded as an errored call.

Example:

1 The following examples show how to change the BER and/or the number of errored seconds. The BERP display changes to reflect the new exponent and the number of errored seconds.

```
> PARMSET BER 6 SECONDS 2
> PARMSET BER 7
> PARMSET SECONDS 2
```

For more information on command syntax and system responses refer to PARMSET in the ‘BERP commands and responses’ section in this document.

Reviewing the previous test set-up

To review the data entered for a test set up, use the REVIEW command. The information includes the list of IBERTs, the list of LSGs, DLCs, trunks, links, the test start and stop times, the length of each test call, the inter-call delay, errored calls or all calls, and the output file location.

Example:

1 The following examples show how to enter the REVIEW command.

```
> REVIEW
```

or

```
> 3
```

For more information on command syntax and system responses refer to REVIEW in the ‘BERP commands and responses’ section in this document.

Selecting the IBERTs

To select the IBERTs to be used in the test, use the SELECT command. The user can specify the command in many ways. The acceptable parameters are ADD, REMOVE, CLEAR, ALL and QUERY.

1 The ADD parameter along with the IBERT type and location adds an IBERT for BERP testing. The IBERT type is either a DTU or an ILC.

2 The REMOVE parameter along with the IBERT type and location removes the specified IBERT from BERP testing.

3 The ALL parameter reserves every IBERT of a specified type for use by the BERP test.

4 The CLEAR parameter removes all IBERTs of a specified type from BERP testing.

5 The QUERY parameter is used to display all the IBERTs reserved for use by BERP.

To obtain a list of which IBERT line cards are present for use in the switch, issue the command POST CARD 6X99AA PRINT . from a hard copy device at the LTP level of the MAP. 0

Note: IBERT cards SELECTed for use in a BERP test are NOT available for Datapath Maintenance.

Examples:

1 The following example shows how to select an IBERT line card (ILC) for the BERP test.

```
> SELECT ADD L HOST 0 1 8 8
```

2 The following example shows how to remove a DTU IBERT from BERP testing.

```
> SELECT REMOVE DTU 0
```

3 The following example shows how to select all IBERT line cards in the switch.

```
> SELECT ALL ILC
```

4 The following example shows how to clear all selected DTU IBERTs from the internal table.

```
> SELECT CLEAR DTU
```

5 The following example shows how to get a list of all IBERTs reserved for use.

```
> SELECT QUERY
```

For more information on command syntax and system responses refer to SELECT in the ‘BERP commands and responses’ section in this document.

Defining the endpoints

To define which endpoints are to be used, use the DEFINE command. The options available for this command are ADD, REMOVE, CLEAR and QUERY.

1 The ADD parameter along with the endpoint type and location, adds the specified endpoint to a pool for BERP testing.

2 The REMOVE parameter along with the endpoint type and location, removes the specified endpoint from testing.

3 The QUERY parameter displays all endpoints defined for testing.

A pool of DLCs and trunks can be specified with this command. The pool must contain at least as many DLCs and trunks as IBERTs. If more DLCs and trunks than IBERTs are specified, the IBERTs connect to the endpoints on a rotating basis. For example, if five IBERT cards have been SELECTed, and five DLCs have been DEFINEd in a pool with five trunks, the first set of test calls connnect the five IBERT cards to the first five DLCs. The next set of test calls connect the five IBERT cards to the second group of five trunks.

Note: LINKs cannot be defined for testing in the same pool as DLCs and trunks. The pool of DLCs and trunks must be cleared before LINKs can be defined for BERP testing.

LSGs cannot be defined for testing in the same pool as DLCs, trunks, or links. The pool of DLCs, trunks and links must be cleared before LSGs can be defined for the test.

All the LINKs on a specified PM can be tested by defining only the pm_type and pm_no. Links must have at least one datafilled trunk in order to be valid for testing.

For LCME line equipment number (lens), the DGRP parameter specifies the digroup tested by BERP. Valid digroup numbers are 0, 1 or 2.

The parameter ISLC specifies an ISDN line card. This parameter is valid for line cards such as NTBX27AA with a 2B1Q line format.

Examples:

1 The following example shows how to define a DLC for testing.

```
> DEFINE ADD L HOST 0 1 8 8
```

2 The following example shows how to remove a trunk from the pool.

```
> DEFINE REMOVE TRK DTC 0 0 1
```

3 The following example shows how to define all datafilled links on a peripheral module for testing.

```
> DEFINE ADD LINK RCC 1
```

4 The following example shows how to get a list of all endpoints reserved for testing.

```
> DEFINE QUERY
```

5 The following example shows how to clear all the endpoints from the pool.

```
> DEFINE CLEAR
```

For more information on command syntax and system responses refer to DEFINE in the ‘BERP commands and responses’ section in this document.

Setting test call parameters

To modify the characteristics of the test call durations, use the CALLSET command. The CALLSET command has the following parameters: LENGTH, DELAY, CALLS, ERRORED or ALLCALLS. These are used to set the length of each test call, the delay between test calls, the number of calls to make in the test, and the calls to be traced by BERP. Since they are all optional parameters, they can be specified individually, in pairs or all together (except ERRORED and ALLCALLS). In the latter case, either ERRORED or ALLCALLS is specified.

- The LENGTH may be tailored to be as low as one minute or as long as 255 hours. The length defaults to 10 minutes.
- The DELAY may be anywhere from one second to 255 hours. The delay defaults to 0, which means that there is no delay between calls.
- The CALLS parameter is used to select how many calls to make during the test. This may be as low as one or as high as 32767. Specifying CONT (continuous) as a subparameter of CALLS causes the IBERTs to continuously make calls of the specified length until a stop command is issued, or the stop time is reached.
- The ERRORED parameter is used to specify that only errored calls are to be traced by BERP; the ALLCALLS parameter is used to specify that all calls are to be traced by BERP. The default setting is ERRORED parameter.

Example

1 The following example shows how to set the call data for a length of 1 minute, a delay between calls of 2 minutes, and a total number of 50 test calls.

```
> CALLSET LENGTH 1 MINS DELAY 2 MINS CALLS 50
```

For more information on command syntax and system responses refer to CALLSET in the ‘BERP commands and responses’ section in this document.

Scheduling the tests

To set the start and stop times of the test, use the DEFTIME command. The form of input required is a weekday (SUN, MON, TUE, WED, THU, FRI or SAT), followed by an hour (based on a 24-hour clock) and the minute. The time (start or stop) is the next occurrence of the input weekday at the specified hour and minute. If a start and stop time are being defined, a check is performed to ensure that the start time is earlier than the stop time. Tests may only be scheduled one at a time.

Examples:

1 The following example shows how to set a start time for Monday at 11:00 AM.

```
> DEFTIME START SET MON 11 00
```

2 The following example shows how to clear a start time.

```
> DEFTIME START CLEAR
```

3 The following example shows how to set a stop time for Tuesday at 7:30 AM.

```
> DEFTIME STOP SET TUE 7 30
```

4 The following example shows how to clear a stop time.

```
> DEFTIME STOP CLEAR
```

5 The following example shows how to clear both the start and stop time.

```
> DEFTIME CLEAR
```

For more information on command syntax and system responses refer to DEFTIME in the ‘BERP commands and responses’ section in this document.

Defining the output file

To define the output file name and file device, use the OUTPUT command. Basic checking to make sure the file device exists is performed. Another check is made to ensure a file with the same name does not already exist on the specified output device. The system allows more than one file on a device to have the same name. However, when attempting to process the file, it does not guarantee that the correct file is used. The file name given must be unique to the output device. The file information can also be cleared with the optional CLEAR parameter. Note that this does not erase the previous file. If an output file is not specified, an errored path data file is not created, and processing of the results is not performed.

Examples:

1 The following example shows how to define a file named BERPFILE and a device named SFDEV.

```
> OUTPUT DEFINE BERPFILE SFDEV
```

2 The following example shows how to clear the file name and device.

```
> OUTPUT CLEAR
```

For more information on command syntax and system responses refer to OUTPUT in the ‘BERP commands and responses’ section in this document.

Checking data for consistency

To ensure that all the data makes sense as a test definition, use the CHECK command when all the data for a test has been input. CHECK verifies that the start and stop times, if defined, are in the proper order and have not already passed, that the IBERTs defined are present, and that the number of endpoints are equal to or greater than the number of IBERTs.

It also checks that an output file has been specified. If an output file has not been specified, a warning is output to indicate that an errored path file will not be created. Also checked are the stop criteria for a test. A test can be set up either to end at a specified time, or after a user-specified number of calls have been made, but not both.

When the CHECK command is entered, all unidentifiable circuits are removed from BERP testing. Any data that is inconsistent is output with an error message. Note that all data is checked at this time and many messages may appear. Refer to 'BERP commands and responses' section in this document for an explanation of the messages. Checking does not stop with the first error encountered. After the data has been verified as correct, the test status is elevated to CHECKED.

Example:

1 The following examples show how to enter the CHECK command.

```
> CHECK
```

or

```
> 8
```

For more information on command syntax and system responses refer to CHECK in the 'BERP commands and responses' section in this document.

Starting a test

To initiate a test, use the START command. This command can only be issued on a test with a status of CHECKED or UNCHECKED.

All the data is verified even if the status is CHECKED, since a great amount of time may have passed since the CHECK command was issued.

If all data is successfully verified, then a test request is issued. If no start time was specified, the test request is to start the test right away. The test status at this point is RUNNING. If a start time is not specified, then a request is issued to start the test at the requested time. The test status at this point is set to WAITING. If a test is in the WAITING state, the STOP command can be used to clear the request to start.

Example:

1 The following example shows how to enter the START command.

```
> START
```

or

```
> 10
```

For more information on command syntax and system responses refer to START in the ‘BERP commands and responses’ section in this document.

Stopping a test

To stop a RUNNING or WAITING test, use the STOP command. This command causes the tests to terminate immediately. Any calls that are in progress are dropped and their data is not included in the test results.

Example:

1 The following example shows how to enter the STOP command.

```
> STOP
```

or

```
> 11
```

For more information on command syntax and system responses refer to STOP in the ‘BERP commands and responses’ section in this document.

Sorting path information

A BERP test call can be considered as a path through a linked series of nodes. The path through each individual node depends on the type of the node. Consequently, the output format for the path trace components of each nodetype differs. The SORTKEY command provides the ability to choose the node type and the path trace variables for sorting the path information. It has the following parameters:

- ADDKEY includes the nodetype and sortkeys for processing. If no sortkey is specified, the default sortkeys for the specified nodetype are used.
- DELKEY deletes the sortkeys of the specified nodetype.
- ADDNODE includes the specified nodetype with default sortkeys.
- DELNODE deletes the specified nodetype.
- ALL includes all nodetypes with their default sortkeys.
- CLEAR removes all nodetypes.

- QUERY queries sortkey for a specific nodetype or all nodetypes.

Example:

1 The following example shows how to enter the SORTKEY command for all nodetypes with default sortkeys.

```
> SORTKEY ALL
```

For more information on command syntax and system responses refer to SORTKEY in the ‘BERP commands and responses’ section in this document.

Processing the test results

To process the errored path file and generate the output reports, use the PROCESS command.

The errored path file created by a BERP test must be processed to generate any reports. The path data file is in machine readable form only.

The process command provides:

1 the header information

2 a test summary (same as the one available from the SUMMARY command)

3 a report for each IBERT and circuit used, with the following information:

   — the number of calls made

   — number of errored calls

   — number of seize failures

   — number of call failures

   — number of interrupted calls

4 sorted path information

The FORMAT option creates a detailed listing of each errored call and its associated path.

Example:

1 The following example shows how to enter a typical PROCESS command with the following parameters:

   — an input filename of BERPF on an input file device called SFDEV

   — an output file name of RESULTS on the same file device called SFDEV

- — the ERRORED option specifies that if non-errored calls are included in the input file, they should be ignored
- — the WORST option specifies the number of worst components (with an errored call ratio worse than 1/1000) to output, in this case 10
- — the CARDS option specifies the number of worst cards (determined by the ratio of errored calls to total calls made through a particular card) to output, in this case 10
- — the FORMAT option to generate a detailed listing of errored calls and paths

```
> PROCESS BERPF SFDEV RESULTS SFDEV ERRORED WORST 10
CARDS 10 FORMAT
```

For more information on command syntax and system responses refer to PROCESS in the ‘BERP commands and responses’ section in this document.

Getting a summary of results

To get a summary of the last known test results, use the SUMMARY command. These values are cleared when a new test is defined. If the test status is STOPPED, the reason the last test was stopped is also output. If a file error occurred while writing to the output file, then an explanation of the type of error encountered is also output.

Example:

1 The following example shows how to enter the summary command.

```
> SUMMARY
```

or

```
> 4
```

For more information on command syntax and system responses refer to SUMMARY in the ‘BERP commands and responses’ section in this document.

Resetting the test data

To reset all the test values back to zero and default at any time, use the RESET command. The following values are reset:

- number of calls made
- errored calls
- error-free calls
- failures to set up calls

- no sync found calls
- calls with a BER worse than the value specified by the user
- calls with more errored seconds than the value specified
- number of interrupted calls
- number of sync slip calls

The ALL parameter may be used to clear all test set-up data, such as the specified IBERTs, DLCs, trunks, links, loopbacks, BER exponent, and the number of errored seconds. The RESET ALL command prompts for a yes or no verification before resetting the test set-up data.

Examples:

1 The following example shows how to reset the test results.

```
> RESET
```

or

```
> 12
```

2 The following example shows how to reset all the test set-up data.

```
> RESET ALL
```

For more information on command syntax and system responses refer to RESET in the ‘BERP commands and responses’ section in this document.

Leaving the BERP level

To leave the BERP level and return to the MTC level of the MAP, use the QUIT command. This is the standard quit command found at all MAP levels. None of the test data is lost by issuing the quit command. Any test in the RUNNING state is left running. Use the QUIT ALL command to return to the command interpreter level.

Examples:

1 The following example shows how to quit from the BERP level and return to the MTC level of the MAP.

```
> QUIT
```

2 The following example shows how to quit from all levels of the MAP and return to the command interpreter (CI).

```
> QUIT ALL
```

For more information on command syntax and system responses refer to QUIT in the ‘BERP commands and responses’ section in this document.

Map displays and output files

Map displays

The key statistics generated by the BERP test calls are as follows:

- the number of calls made
- the number of errored calls
- calls with a Bit Error Ratio greater than the value specified by the user
- calls with a greater number of errored seconds than specified by the user
- While BERP tests are running, these values are updated in real time at the BERP level of the MAP. When tests are not running, the last known test statistics are displayed along with an indication that tests are not currently running.

Output files

BERP output is produced by the PROCESS command. It takes an input file and directs the output to an output file, identified by the specified output file device and file name. The PROCESS command can be used to generate the following three components of the output file:

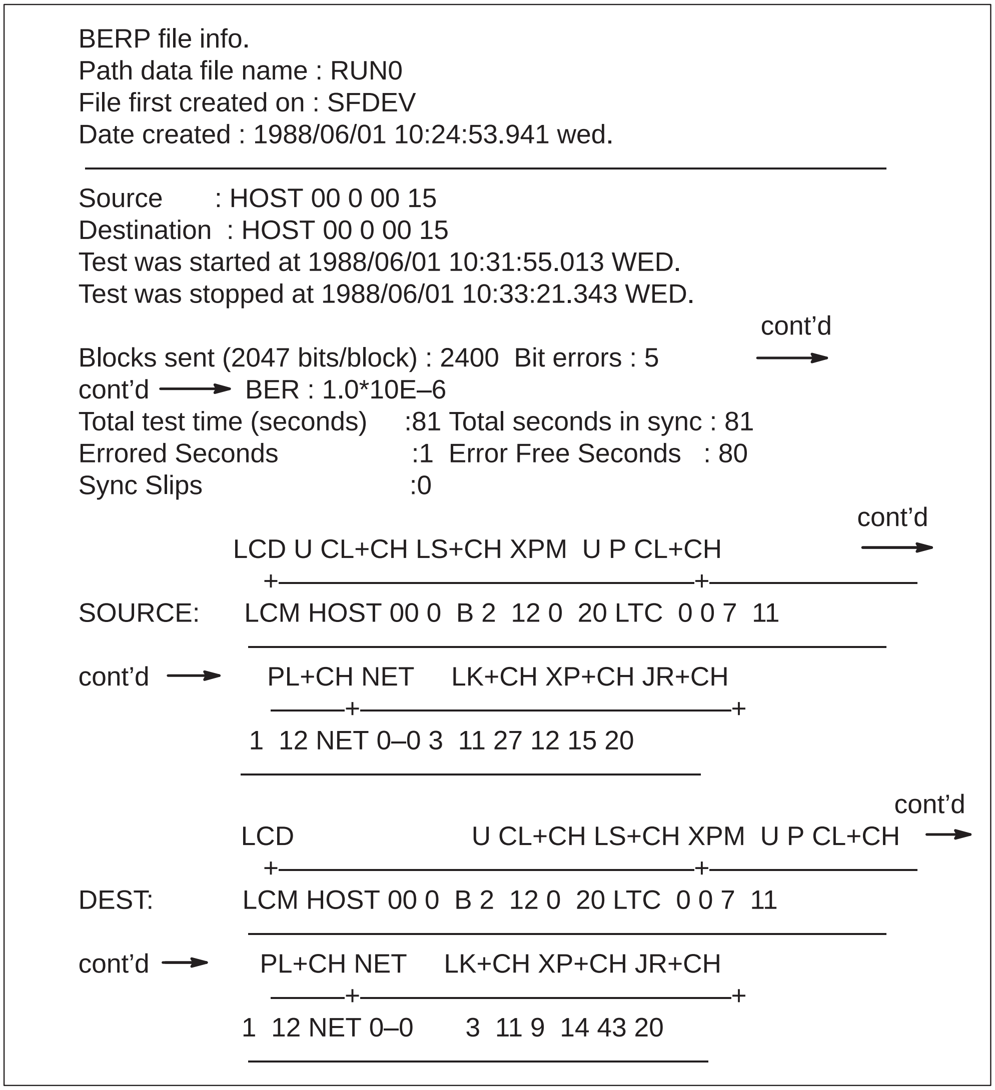
- a complete path trace of each errored call
- a summary of the BERP run
- the sorted path information

The first part (path trace of each errored call) of the output file is output only with the FORMAT option of the PROCESS command. The last part (sorted path information) is output depending on the nodetypes and path components defined by the SORTKEY command.

A sample of a processed output file, when the PROCESS command is used with FORMAT option, is shown in Figure 4-1 on page 4-2.

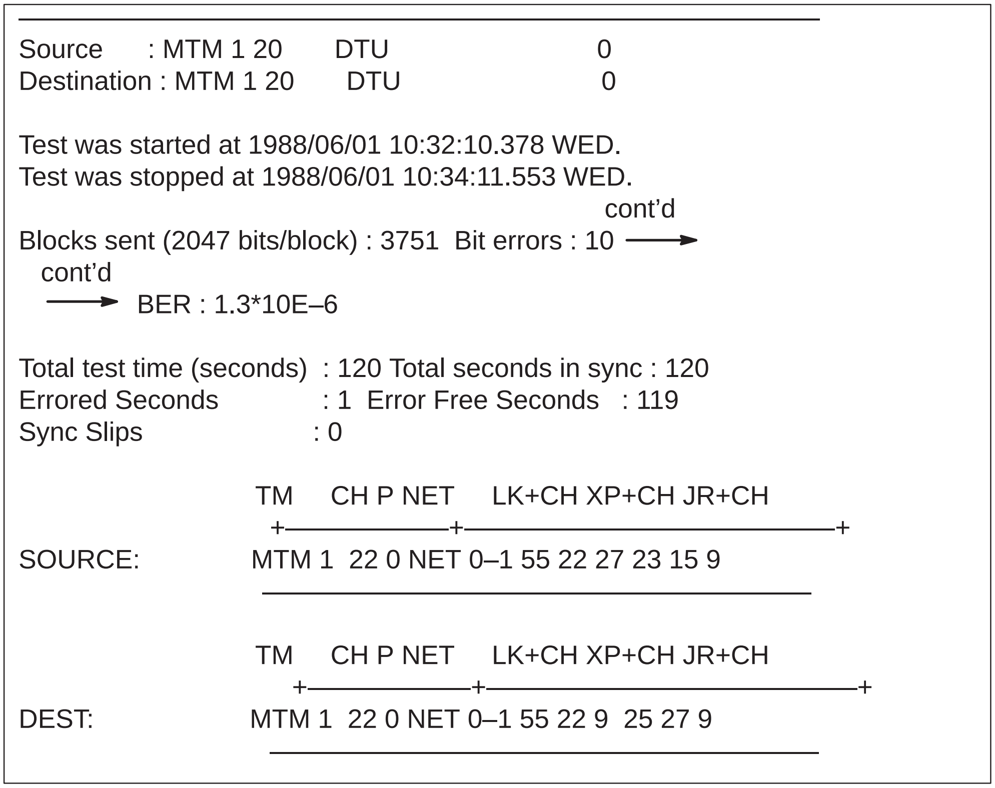
Note: Since the information in a BERP output file is presented in a wider format than the pages of this manual allow, some of the output information is continued below the top portion of the figure.

Figure 4-1
Sample (FORMAT Option) BERP Output



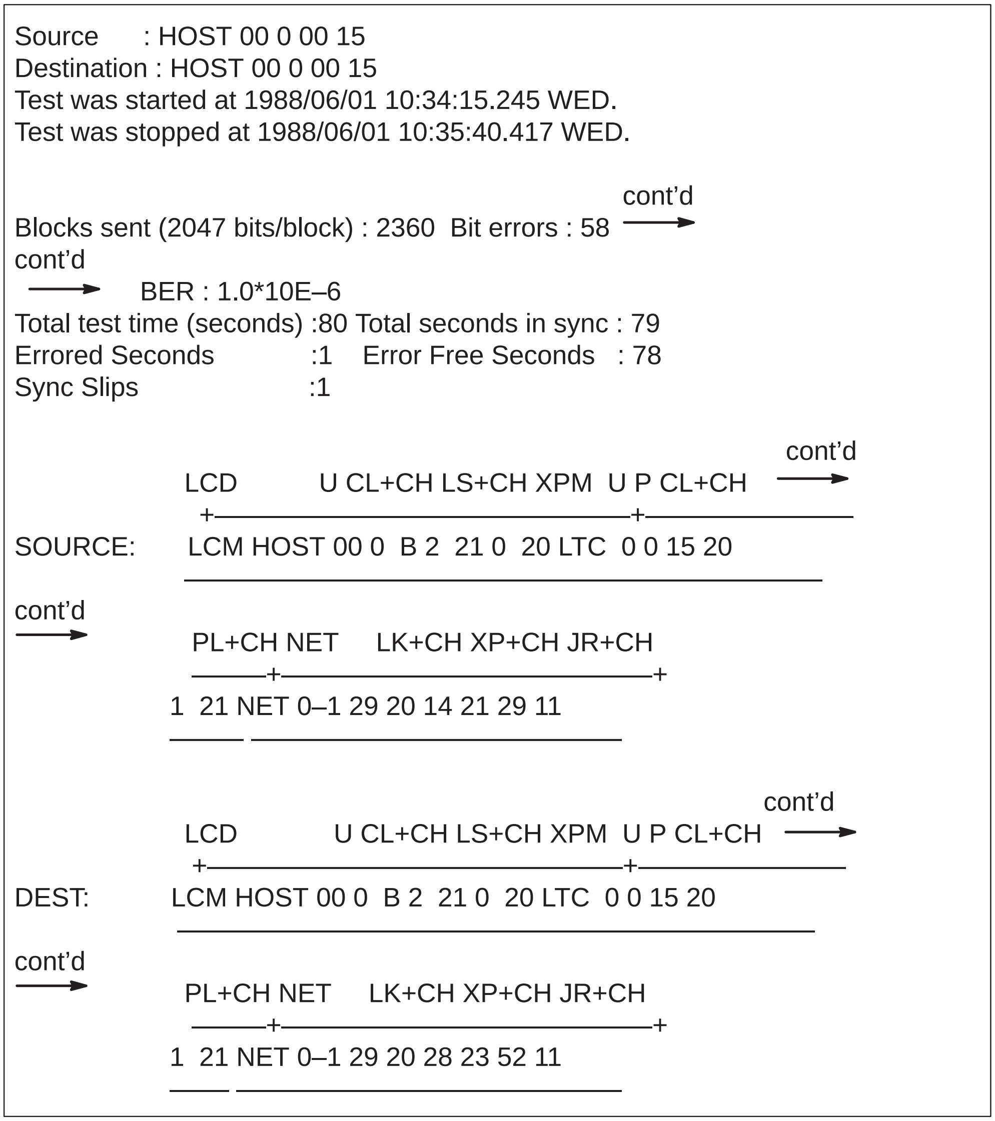
```
BERP file info.
Path data file name : RUN0
File first created on : SFDEV
Date created : 1988/06/01 10:24:53.941 wed.
────────────────────────────────────────────────────────
Source       : HOST 00 0 00 15
Destination  : HOST 00 0 00 15
Test was started at 1988/06/01 10:31:55.013 WED.
Test was stopped at 1988/06/01 10:33:21.343 WED.
                                                         cont’d
Blocks sent (2047 bits/block) : 2400  Bit errors : 5      ──→
cont’d ──→ BER : 1.0*10E–6
Total test time (seconds)    :81 Total seconds in sync : 81
Errored Seconds               :1  Error Free Seconds   : 80
Sync Slips                    :0
                                                                cont’d
            LCD U CL+CH LS+CH XPM  U P CL+CH                    ──→
              +──────────────────────────────+──────────────
SOURCE:      LCM HOST 00 0  B 2  12 0  20 LTC  0 0 7  11
             ──────────────────────────────────────────────
cont’d ──→     PL+CH NET     LK+CH XP+CH JR+CH
               ─────+─────────────────────────+
             1  12 NET 0–0 3  11 27 12 15 20
            ────────────────────────────────
                                                                  cont’d
            LCD                  U CL+CH LS+CH XPM  U P CL+CH  ──→
              +──────────────────────────────+──────────────
DEST:        LCM HOST 00 0  B 2  12 0  20 LTC  0 0 7  11
             ──────────────────────────────────────────────
cont’d ──→    PL+CH NET     LK+CH XP+CH JR+CH
               ─────+─────────────────────────+
            1  12 NET 0–0       3  11 9  14 43 20
             ────────────────────────────────
```

Figure 4-2
Sample (FORMAT Option) BERP Output (Part 2 of 6)



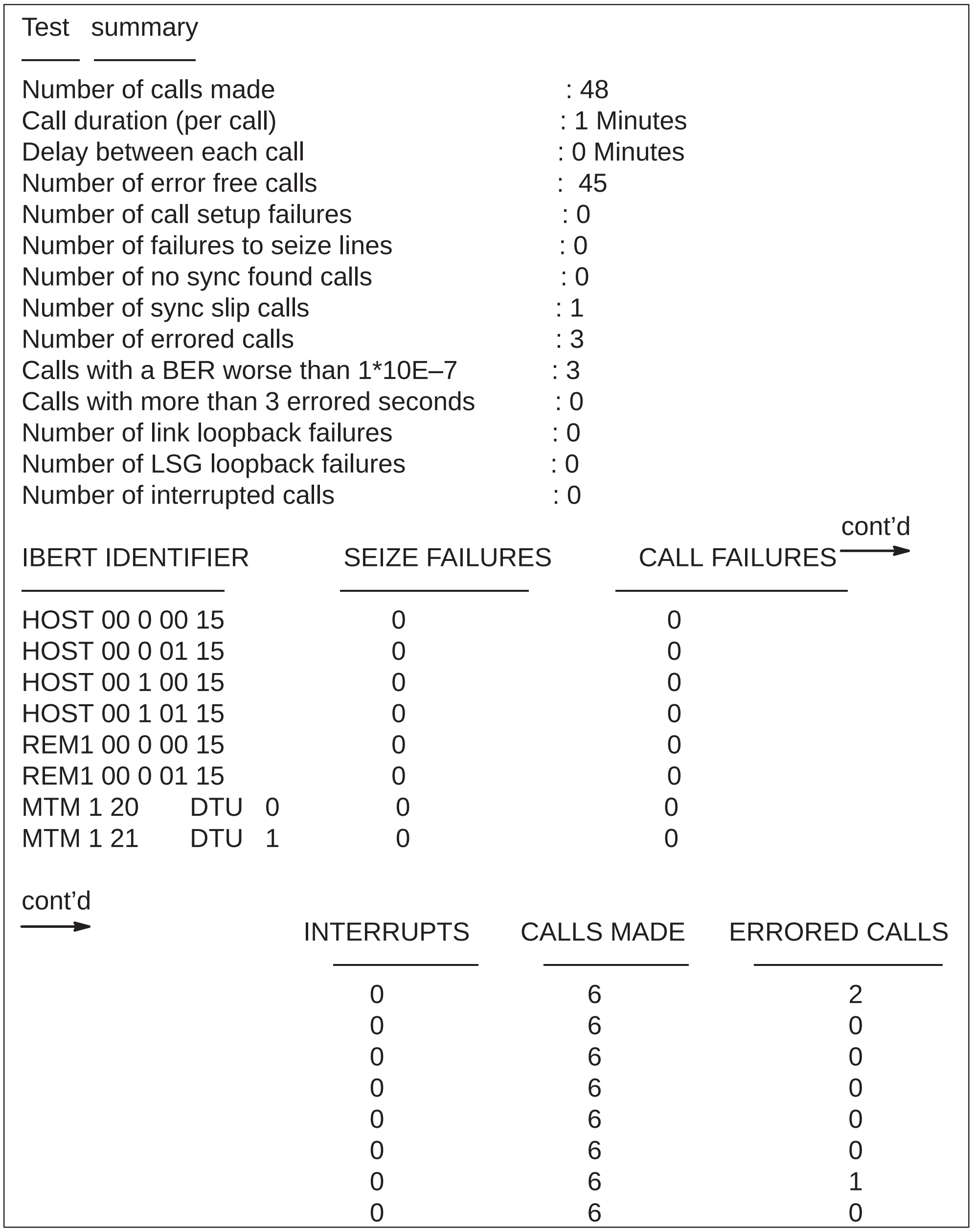
```
____________________________________________________________________
Source      : MTM 1 20       DTU                          0
Destination : MTM 1 20       DTU                          0

Test was started at 1988/06/01 10:32:10.378 WED.
Test was stopped at 1988/06/01 10:34:11.553 WED.
                                                          cont'd
Blocks sent (2047 bits/block) : 3751  Bit errors : 10 ------>
  cont'd
  ------>  BER : 1.3*10E–6

Total test time (seconds)  : 120 Total seconds in sync : 120
Errored Seconds            : 1  Error Free Seconds   : 119
Sync Slips                 : 0

                    TM     CH P NET     LK+CH XP+CH JR+CH
                     +---------------+------------------------------+
SOURCE:             MTM 1  22 0 NET 0–1 55 22 27 23 15 9
                     ______________________________________________


                    TM     CH P NET     LK+CH XP+CH JR+CH
                       +---------------+------------------------------+
DEST:               MTM 1  22 0 NET 0–1 55 22 9  25 27 9
                      ______________________________________________
```

Figure 4-3
Sample (FORMAT Option) BERP Output (Part 3 of 6)



```
Source      : HOST 00 0 00 15
Destination : HOST 00 0 00 15
Test was started at 1988/06/01 10:34:15.245 WED.
Test was stopped at 1988/06/01 10:35:40.417 WED.

                                                   cont'd
Blocks sent (2047 bits/block) : 2360  Bit errors : 58  →
cont'd
  →         BER : 1.0*10E–6
Total test time (seconds) :80 Total seconds in sync : 79
Errored Seconds           :1    Error Free Seconds   : 78
Sync Slips                :1

                                                                   cont'd
               LCD          U CL+CH LS+CH XPM  U P CL+CH   →
                +————————————————————————————————+——————————————
SOURCE:        LCM HOST 00 0  B 2  21 0  20 LTC  0 0 15 20
               ————————————————————————————————————————————————
cont'd
  →             PL+CH NET     LK+CH XP+CH JR+CH
                ——————+————————————————————————————+
              1  21 NET 0–1 29 20 14 21 29 11
              —————— ————————————————————————————

                                                                 cont'd
               LCD            U CL+CH LS+CH XPM  U P CL+CH   →
                +————————————————————————————————+——————————————
DEST:         LCM HOST 00 0  B 2  21 0  20 LTC  0 0 15 20
              —————————————————————————————————————————————————
cont'd
  →            PL+CH NET     LK+CH XP+CH JR+CH
                ——————+————————————————————————————+
              1  21 NET 0–1 29 20 28 23 52 11
              ————— ——————————————————————————————
```

Figure 4-4
Sample (FORMAT Option) BERP Output (Part 4 of 6)



```
Test   summary
____   _______

Number of calls made                      : 48
Call duration (per call)                  : 1 Minutes
Delay between each call                   : 0 Minutes
Number of error free calls                :  45
Number of call setup failures             : 0
Number of failures to seize lines         : 0
Number of no sync found calls             : 0
Number of sync slip calls                 : 1
Number of errored calls                   : 3
Calls with a BER worse than 1*10E–7       : 3
Calls with more than 3 errored seconds    : 0
Number of link loopback failures          : 0
Number of LSG loopback failures           : 0
Number of interrupted calls               : 0
                                                                 cont’d
IBERT IDENTIFIER         SEIZE FAILURES        CALL FAILURES ──────>
________________         ______________        _____________

HOST 00 0 00 15               0                     0
HOST 00 0 01 15               0                     0
HOST 00 1 00 15               0                     0
HOST 00 1 01 15               0                     0
REM1 00 0 00 15               0                     0
REM1 00 0 01 15               0                     0
MTM 1 20     DTU   0          0                     0
MTM 1 21     DTU   1          0                     0

cont’d
──────>
                      INTERRUPTS      CALLS MADE      ERRORED CALLS
                      __________      __________      _____________

                           0               6                2
                           0               6                0
                           0               6                0
                           0               6                0
                           0               6                0
                           0               6                0
                           0               6                1
                           0               6                0
```

Figure 4-5
Sample (FORMAT Option) BERP Output (Part 5 of 6)

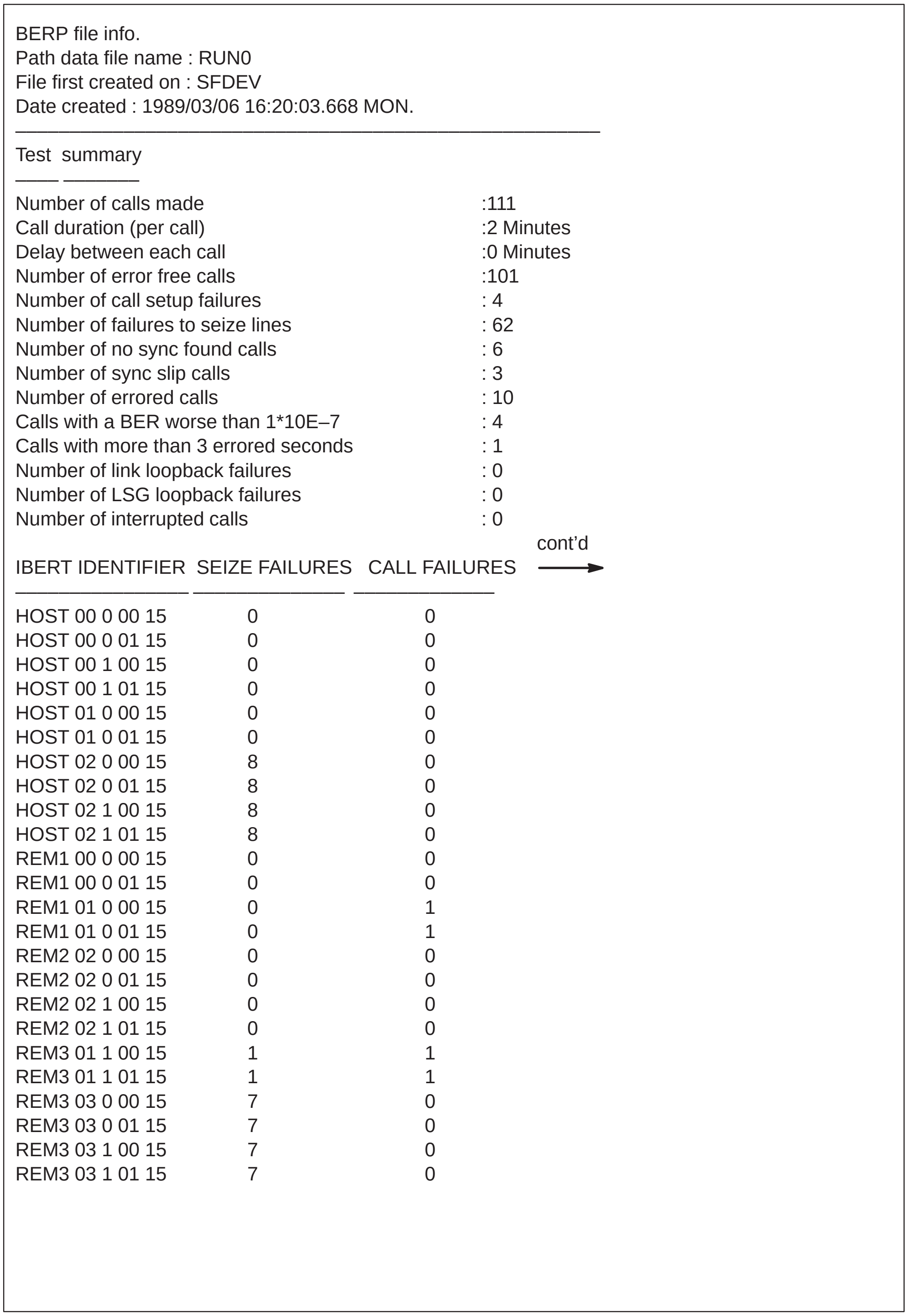
```
Sorted Path Information:
 ____________________________

NET     LK   ERRORED  NO SYNC  BIT ERRORS
____________  ________________________________

net 0–0 3       2         0            10
               ________________________________

                2         0            10

NET     JR   ERRORED  NO SYNC  BIT ERRORS
 ____________  ________________________________

net 0–0 15     1          0             5
net 0–0 43     1          0             5
               ________________________________

               2          0            10

NET     LK   ERRORED  NO SYNC  BIT ERRORS
____________  ________________________________

net 0–1 29     2          0           116
net 0–1 55     2          0            20
               ________________________________

               4          0           136

NET     JR   ERRORED  NO SYNC  BIT ERRORS
 ____________  ________________________________

 net 0–1 15     1          0            10
 net 0–1 27     1          0            10
 net 0–1 29     1          0            58
 net 0–1 52     1          0            58
                ________________________________

               4          0            136
 TM      CH   ERRORED  NO SYNC  BIT ERRORS
 ____________  ________________________________

 MTM 1   22    1           0            10
                ________________________________

               1           0             10
```

Figure 4-6
Sample (FORMAT Option) BERP Output (Part 6 of 6)

```
XPM     PL   ERRORED  NO SYNC  BIT ERRORS
-----------  ----------------------------
LTC 0    1      2        0         63
              ----------------------------
               2         0         63

XPM     CL   ERRORED  NO SYNC  BIT ERRORS
-----------  ----------------------------
LTC 0    7      1        0          5
LTC 0   15      1        0         58
             ----------------------------
               2         0         63

LCD                LS   ERRORED  NO SYNC  BIT ERRORS
-------------------    ----------------------------
LCM HOST 00  0   0        2         0          63
                       ----------------------------
                         2         0          63

LCD                CL   ERRORED  NO SYNC  BIT ERRORS
-------------------    ----------------------------
LCM HOST 00  0   2        2         0          63
                       ----------------------------
                         2         0          63
```

A sample of a processed output file, without the FORMAT option and with all calls traced, is shown in Figure 4-7 on page 4-8. Only a portion of the complete output is shown in this example in order to provide a sample of the structure of the report.

Figure 4-7
Sample (No FORMAT Option) BERP Output (Part 1 of 3)

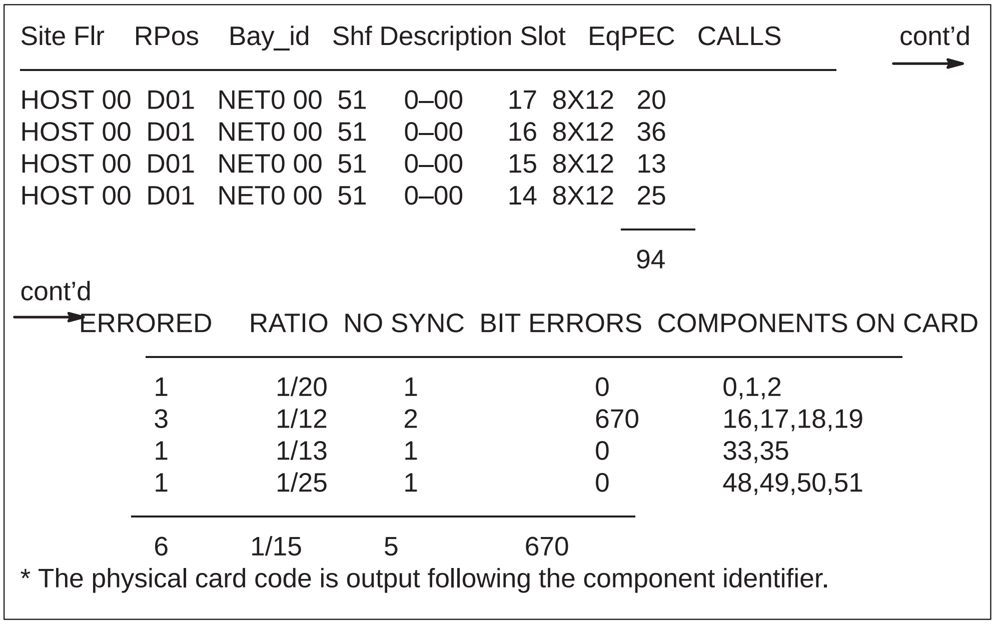


```
BERP file info.
Path data file name : RUN0
File first created on : SFDEV
Date created : 1989/03/06 16:20:03.668 MON.
————————————————————————————————————————————————
Test  summary
——— —————
Number of calls made                            :111
Call duration (per call)                        :2 Minutes
Delay between each call                         :0 Minutes
Number of error free calls                      :101
Number of call setup failures                   : 4
Number of failures to seize lines               : 62
Number of no sync found calls                   : 6
Number of sync slip calls                       : 3
Number of errored calls                         : 10
Calls with a BER worse than 1*10E–7             : 4
Calls with more than 3 errored seconds          : 1
Number of link loopback failures                : 0
Number of LSG loopback failures                 : 0
Number of interrupted calls                     : 0
                                                      cont'd
IBERT IDENTIFIER  SEIZE FAILURES   CALL FAILURES   ——→
————————————————  ——————————————   —————————————
HOST 00 0 00 15           0               0
HOST 00 0 01 15           0               0
HOST 00 1 00 15           0               0
HOST 00 1 01 15           0               0
HOST 01 0 00 15           0               0
HOST 01 0 01 15           0               0
HOST 02 0 00 15           8               0
HOST 02 0 01 15           8               0
HOST 02 1 00 15           8               0
HOST 02 1 01 15           8               0
REM1 00 0 00 15           0               0
REM1 00 0 01 15           0               0
REM1 01 0 00 15           0               1
REM1 01 0 01 15           0               1
REM2 02 0 00 15           0               0
REM2 02 0 01 15           0               0
REM2 02 1 00 15           0               0
REM2 02 1 01 15           0               0
REM3 01 1 00 15           1               1
REM3 01 1 01 15           1               1
REM3 03 0 00 15           7               0
REM3 03 0 01 15           7               0
REM3 03 1 00 15           7               0
REM3 03 1 01 15           7               0
```

Figure 4-8
Sample (No FORMAT Option) BERP Output (Part 2 of 3)

```
cont’d
───►       INTERRUPTS  CALLS MADE  ERRORED CALLS
           __________  __________  _____________
                0           8            1
                0           8            0
                0           8            0
                0           8            0
                0           8            0
                0           8            1
                0           0            0
                0           0            0
                0           0            0
                0           0            0
                0           7            0
                0           7            0
                0           6            1
                0           6            0
                0           7            0
                0           7            0
                0           7            0
                0           7            0
                0           5            4
                0           4            3
                0           0            0
                0           0            0
                0           0            0
               0            0            0
SORTED PATH INFORMATION
_______________________
NET     LK CARD*    CALLS  ERRORED  RATIO  NO SYNC  BIT ERRORS
_______________     __________________________________________
NET 0–0  0 8X12        1        0     0/1       0          0
NET 0–0  1 8X12       18       1     1/18       1          0
NET 0–0  2 8X12        1       0      0/1       0          0
NET 0–0 16 8X12        4       0      0/4       0          0
NET 0–0 17 8X12       25       2     1/12      1         670
NET 0–0 18 8X12        3       0      0/3      0           0
NET 0–0 19 8X12        4       1      1/4      1           0
NET 0–0 33 8X12       12       1     1/12      1           0
NET 0–0 35 8X12        1       0       0/1     0           0
NET 0–0 48 8X12        3       0       0/3     0           0
NET 0–0 49 8X12       15       1     1/15      1           0
NET 0–0 50 8X12        4       0       0/4     0           0
NET 0–0 51 8X12        3       0       0/3     0           0
                    ___________________________________________
                     94         6    1/15       5         670
```

Figure 4-9
Sample (No FORMAT Option) BERP Output (Part 3 of 3)



```
Site Flr   RPos   Bay_id  Shf Description Slot  EqPEC  CALLS          cont'd
                                                                       ------>
HOST 00  D01   NET0 00  51    0–00     17  8X12   20
HOST 00  D01   NET0 00  51    0–00     16  8X12   36
HOST 00  D01   NET0 00  51    0–00     15  8X12   13
HOST 00  D01   NET0 00  51    0–00     14  8X12   25
                                                 ----
                                                  94
cont'd
---->ERRORED     RATIO  NO SYNC  BIT ERRORS  COMPONENTS ON CARD

          1          1/20      1                0          0,1,2
          3          1/12      2                670        16,17,18,19
          1          1/13      1                0          33,35
          1          1/25      1                0          48,49,50,51
        ------------------------------------------------
          6       1/15        5            670
* The physical card code is output following the component identifier.
```

Notes:

1 The last report shown in part 3 of Figure 4-9 on page 4-10 summarizes the component report on a per card basis in order to indicate which physical cards experienced errors. This type of report is produced for each of the sortkeys on each node type to be output. In certain instances the components do not correspond to physical cards. For example, no physical junctor cards exist for networks which contain short junctors. In this case the card field on the component report is set to N/A. Also, no card report is produced since no physical card exists.

2 The c-side links of an LCM are connected through the 6X52 Digroup Control Card (DCC), while the c-side links of an RLCM are connected through a 6X50 (DS-1 links). As a result, the report on physical cards which follows the sortkey report on c-side links will be based on 6X52 cards in the case of LCMs and on 6X50 cards for the RLCM.

Errored path breakdown

Each errored path breakdown contains all the information concerning the transmission path taken. Typically, calls are set up from one network plane to another when both networks are in service (excluding IBERT to IBERT calls). In this case both ends of the call transmit on both planes, but receive only on one plane. The planes that are output in the path breakdown are the planes that the end was receiving on. These path breakdowns are produced only when the FORMAT option is used with the PROCESS command (refer to PROCESS in the ‘BERP commands and responses’ section in this document).

Note: In some path traces a dash (-) appears in place of a p-side link number for either of the following reasons:

1 The PM does not have a p-side link. This applies only to MTMs.

2 A DS30 loopback was used to test a trunk. With a DS30 loopback, the test call does not pass through the p-side of the PM. The processed output file helps identify the faulty components of the faulty paths. Once these are identified, they may be tested using some of the other tools such as NETPATH or ICTS for network components, or XBERT for LTC/LCM components. These tools can be used to diagnose specific sections of the paths which appear to be faulty, and isolate the trouble area down to either a single faulty card or a list of potentially faulty cards. More information on these tools can be found in Additional Test Tools in the ‘Test description’ section in this document.

BERP commands and responses

Commands and responses

The commands, parameters, and responses described in this chapter are in the order that they appear on the BERP menu.

The BERP menu commands appear in the following order:

- BERP
- 0 Quit_
- 2
- 3 Review
- 4 Summary
- 5 Select_
- 6 Define_
- 7 Callset_
- 8 Deftime_
- 9 Check
- 10 Start
- 11 Stop
- 12 Output_
- 13 Reset_
- 14 Parmset_
- 15 Sortkey_
- 16 Process_
- 17 Loopbk_

Full details of command input methods can be found in NTP 297-1001-520. If a situation arises where you can not quit from a BERP command, simply enter ABORT to quit from the command.

BERP	

BERP

This command is used to enter the BERP level of the MAP.

Responses
`The MAP level for BERP replaces the MTC level display.` **Meaning:** Commands may now be used to set up system-wide Bit Error Rate Performance (BERP) tests to assess the performance of the DMS-100 Family switch. **System Action:** None **User Action:** Set up the BERP test

Usage Note:

No special characteristics apply to this command.

Examples:

1 The following example shows how to access the BERP level from the command interpreter (CI) level of the MAP.

> MAPCI;MTC;BERP

2 The following examples show how to access the BERP level from the MTC level of the MAP.

> 5

or

> BERP

or

> berp

QUIT	[ALL]

QUIT

This command is used to exit the BERP level of the MAP. QUIT may be issued while a test is RUNNING.

where:

ALL May be used to exit all levels of the MAP and return to the Command Interpreter (CI).

Responses
`The MAP level display for MTC replaces the BERP level of the display.` **Meaning:** Commands for the MTC level of the MAP can be issued. **System Action:** None **User Action:** Enter the desired command from the MTC level of the MAP.

Usage Note:

None of the test data is lost by issuing the QUIT command. Any test in the RUNNING state is left running.

Examples:

1 The following example shows how to QUIT from the BERP level and return to the MTC level of the MAP.

> QUIT

2 The following example shows how to quit from all levels of the MAP and return to the Command Interpreter (CI).

> QUIT ALL

REVIEW	

REVIEW

This command is used to review all test set-up information for the BERP test. The information displayed is:

- The current test status.
- Length of each call.

- Length of delay between calls.
- How many test calls are to be made.
- How many IBERT cards will be used in the test and their physical location in the switch.
- How many DLCs or trunks will be tested and their location in the switch.
- The links defined for testing and which loopback is to be used (DS30, DS1 or none).
- The start time of the test.
- The stop time of the test.
- The output file to be used.

(sample)

```
TESTS ARE CURRENTLY STOPPED

CALL LENGTH IS SET AT : 10 MINUTES
DELAY BETWEEN CALLS IS SET AT : 0 MINUTES

A MINIMUM OF 3 CALLS ARE TO BE MADE.

YOU HAVE SELECTED 1 IBERTS FOR USE, THEY ARE :

HOST 00 0 02 11

4 CIRCUITS HAVE BEEN SELECTED FOR THE IBERTS
TO CONNECT TO. THEY ARE AT :

HOST 00 0 00 07
HOST 00 0 08 05
DTC 0 0 1      OGDP 1
DTC 0 0 2      OGDP 2

NO START TIME HAS BEEN SELECTED.
NO STOP TIME HAS BEEN SELECTED.
```

Responses

```
OUTPUT FILE HAS BEEN DEFINED AS BERPFILE SFDEV.
```

Meaning: A summary of the test set-up information is displayed as in the previous sample.

System Action: None

User Action: Further commands may be issued to change or add new data for the test, or to CHECK the data for consistency.

```
ALL CALLS WILL BE TRACED
```

Meaning: Berp test is set up for all calls to be traced.

System Action: None

User Action: None

```
ERRORED CALLS WILL BE TRACED
```

Meaning: Berp test is set up for only errored calls to be traced.

System Action: None

User Action: None

Example:

1 The following example shows how to enter the REVIEW command.

> REVIEW

or

> 3

SUMMARY	

SUMMARY

This command displays the last known test results (if any). Statistics that will be displayed are :

- Number of calls made.
- Call duration (per call).

- Delay between each call.
- Number of error free calls.
- Number of call set-up failures.
- Number of failures to seize lines.
- Number of no sync found calls.
- Number of sync slip calls.
- Number of errored calls.
- Calls with a BER worse than the value specified by the user.
- Calls with more errored seconds than the value specified by the user.
- Number of link loopback failures.
- Number of LSG loopback failures.
- Number of Interrupted Calls.
- Reason the last test stopped.

Responses

```
TEST SUMMARY
---- ---------
NUMBER OF CALLS MADE                           : 48
CALL DURATION (PER CALL)                       : 1 MINUTES
DELAY BETWEEN EACH CALL                        : 0 MINUTES
NUMBER OF ERROR FREE CALLS                     : 45
NUMBER OF CALL SETUP FAILURES                  : 0
NUMBER OF FAILURES TO SEIZE LINES              : 0
NUMBER OF NO SYNC FOUND CALLS                  : 0
NUMBER OF SYNC SLIP CALLS                      : 1
NUMBER OF ERRORED CALLS                        : 3
CALLS WITH A BER WORSE THAN 1*10E-7            : 3
CALLS WITH MORE THAN 3 ERRORED SECONDS         : 0
NUMBER OF LINK LOOPBACK FAILURES               : 0
NUMBER OF LSG LOOPBACK FAILURES                : 0
NUMBER OF INTERRUPTED CALLS                    : 0
```

—continued—

Responses

```
LAST TEST STOPPED WHEN END CRITERIA WAS MET.
```

Meaning: The test data is summarized.

System Action: None

User Action: None

```
NUMBER OF SYNC SLIP CALLS : n
```

Meaning: The number of sync slip calls is listed.

System Action: None

User Action: None

```
LAST TEST STOPPED DUE TO USER ISSUED STOP COMMAND.
```

Meaning: The last test was stopped by the user.

System Action: None

User Action: None

```
LAST TEST STOPPED DUE TO A FILE ERROR, ’RETURN CODE’.
```

Meaning: The last test stopped due to a file problem. The return code indicates the type of problem encountered.

System Action: None

User Action: Correct the problem identified by the return code.

```
LAST TEST ABORTED DUE TO A DATA ERROR.
ISSUE CHECK COMMAND TO SEE THE ERROR.
```

Meaning: The last test stopped due to a data error.

System Action: None

User Action: Correct the problem identified by the CHECK command.

```
THE FOLLOWING LINKS ON DTC 0 HAD LINK LOOPBACK FAILURES:
LINK 1. REASON: specified
```

—continued—

Responses
`LINK 3. REASON: specified` **Meaning:** The SUMMARY command was entered and the links listed for the specified PM had at least one loopback failure while testing. The reason for the failure is specified for the link. The following is a list of possible reasons: Unequipped - The link is not datafilled. Unavailable - The link is not in a MANB state. PM Busy - The PM is not in service. Failed - Failed to set the loopback. Local Loopback - Link has a local loopback set. Remote Loopback - Link has a remote loopback set. **System Action:** None. **User Action:** Enter the CHECK command to check the status of the links and the loopbacks.
—end—

Usage Note:

The reason the last test was stopped appears only when the test status is STOPPED. The test results for any test are cleared whenever the test status changes from STOPPED. This occurs when any command is issued to change a parameter of a test set up.

Example:

1 The following example shows how to enter the SUMMARY command.

```
> SUMMARY
```

or

> 4

```
 SELECT  |       ADD          DTU   no
         |       REMOVE        L     [site]   len
         |
         |
         |       CLEAR        ckt_type
         |       ALL
         |
         |
         |       QUERY
```

SELECT

This command selects the IBERTs to be used for the test. The IBERTs can be any combination of DTUs and ILCs.

where:

ADD — prepares to add a specific IBERT to an internal table for BERP testing.
REMOVE — prepares to remove a specific IBERT from the internal table.
DTU — indicates that a DTU (in an MTM) is to be specified.
L — indicates that an ILC (in an LCM) is to be specified.
no — is the external number of the DTU from 0 to 9999.
site — is the location of the ILC equipment (such as HOST or REM2 etc.).
len — is the Line Equipment Number (LEN) of the ILC which consists of the string ff u dd cc.

where:

ff — is the frame number from 00 to 99.
u — is the unit number from 0 to 9.
dd — is the drawer number from 00 to 19.
cc — is the circuit number from 00 to 31.

CLEAR — prepares to clear the specified IBERTs from the internal table (specified by ckt_type).
ALL — prepares to select all the specified IBERTs for BERP testing (specified by ckt_type).
ckt_type — specifies the type of IBERT, either DTU or ILC.
QUERY — displays a list of the IBERTs reserved for use.

Responses

```
BERP IS NOT ALLOWED TO ACCESS THIS IBERT
```

Meaning: The selected IBERT cannot be used by BERP because of datafill in tables FMRESUSE and FMRESINV.

System Action: None

User Action: Check if datafill of FMRESUSE and FMRESINV tables can be changed.

```
WARNING - THIS IBERT IS SUSPECTED OF BEING FAULTY OR HAS
FAILED A DIAGNOSTIC.
```

Meaning: The selected IBERT has either been marked suspect or has failed the last diagnostic.

System Action: None

User Action: Run a diagnostic on the IBERT. If it passes the diagnostic, it is functional; if it fails, it is faulty and should be removed by using SELECT REMOVE.

```
THE IBERT IS ALREADY IN THE TABLE.
```

Meaning: The command SELECT ADD was entered and the specified IBERT was already selected for use.

System Action: None

User Action: None

```
NO MORE ROOM TO STORE THIS IBERT.
```

Meaning: The command SELECT ADD was entered and the internal table was full.

System Action: None

User Action: Either remove some of the IBERTs from the table using the SELECT REMOVE or SELECT CLEAR command, or run the test using the currently SELECTed IBERTs.

```
IBERT HAS BEEN RESERVED FOR LATER USE BY BERP.
```

—continued—

Responses

Meaning: The command SELECT ADD was entered and the IBERT was put into the internal table for use.

System Action: None

User Action: None

```
IBERT REMOVED FROM USE BY BERP.
```

Meaning: The command SELECT REMOVE was entered and the IBERT was removed from the internal table.

System Action: None

User Action: None

```
IBERT NOT FOUND, NO ENTRY DELETED.
```

Meaning: The command SELECT REMOVE was entered and the IBERT was not found in the internal table.

System Action: None

User Action: None

```
n IBERTS HAVE BEEN SELECTED, ISSUE REVIEW COMMAND TO SEE
THEM.
```

Meaning: The command SELECT ALL was entered. All IBERTs in the DMS were reserved for use by BERP. The n represents the number of IBERTs reserved.

System Action: None

User Action: None

```
NO IBERTS HAVE BEEN SELECTED FOR USE.
```

Meaning: The command SELECT QUERY was entered and no IBERTs were reserved for use.

System Action: None

User Action: None

```
YOU HAVE SELECTED N IBERTS FOR USE, THEY ARE : ’list of
line equipment numbers or trunk locations and CLLIs’
```

—continued—

Responses

Meaning: The SELECT QUERY command was entered and the list of IBERTs reserved for use was displayed. The n represents the total number of IBERTs reserved for use.

System Action: None

User Action: None

```
INVALID IBERT LINE EQUIPMENT NUMBER ENTERED.
```

Meaning: The SELECT ADD or SELECT REMOVE command was entered for an ILC and the specified LEN was not for an IBERT line card.

System Action: None

User Action: Check the LEN and re-enter the command.

```
INPUT LEN IS NOT FOR AN IBERT LINE CARD.
```

Meaning: The SELECT ADD or SELECT REMOVE command was entered for an ILC and an invalid LEN was specified.

System Action: None

User Action: Check the LEN and re-enter the command.

```
UNKNOWN LINE CARD...INPUT IGNORED.
```

Meaning: The SELECT ADD or SELECT REMOVE command was entered for an ILC and an invalid LEN was specified.

System Action: None

User Action: Check the LEN and re-enter the command.

```
FAILED TO OBTAIN (ADDITIONAL) IBERTS.
```

Meaning: The command SELECT ALL was entered and additional IBERTs could not be obtained.

System Action: None

User Action: None

```
FAILED TO OBTAIN (ADDITIONAL) IBERTs OF REQUESTED TYPE.
```

—continued—

Responses

Meaning: The command SELECT ALL was entered but no more IBERTs of the specified type could be obtained.

System Action: None

User Action: None

```
SELECT FULL. NOT ALL IBERTs COULD BE SELECTED.
```

Meaning: The command SELECT ALL was entered but there was no more room in the internal table. A maximum of 64 IBERTs can be selected.

System Action: None

User Action: None

```
NO CIRCUITS OF REQUESTED TYPE WERE FOUND.
```

Meaning: The SELECT CLEAR command was entered and no IBERT circuits of that type were found.

System Action: None

User Action: None

```
ONE OR MORE UNIDENTIFIABLE CIRCUITS WERE REMOVED.
```

Meaning: The command SELECT CLEAR was entered and some invalid IBERTs were cleared along with the specified IBERTs.

System Action: None

User Action: None

```
CIRCUITS OF REQUESTED TYPE WERE REMOVED.
```

Meaning: The command SELECT CLEAR was entered and all IBERTs of the specified type were removed from the internal table.

System Action: None

User Action: None

```
NO IBERTS HAVE BEEN SELECTED AS TESTERS.
```

—continued—

Responses

Meaning: The command SELECT REMOVE was entered when there were no IBERTs selected for BERP tests.

System Action: None

User Action: None

```
IBERT REMOVED.
```

Meaning: The command SELECT REMOVE was entered and the specified IBERT was removed from the internal table.

System Action: None

User Action: None

```
FAILED TO RESERVE IBERT.
```

Meaning: The SELECT ADD command was entered while the specified IBERT was being used for a BERT test.

System Action: None

User Action: If the IBERT must be selected, stop the BERT test and select the IBERT.

```
IBERT HAS ALREADY BEEN SELECTED AS A BERP TESTER.
```

Meaning: The command SELECT ADD was entered for an IBERT already selected for BERP testing.

System Action: None

User Action: None

```
CIRCUIT IS NOT A VALID IBERT.
```

Meaning: The command SELECT ADD was entered for a circuit that was not an IBERT.

System Action: None

User Action: Check the datafill of the specified circuit.

—continued—

Responses
`UNIDENTIFIABLE CIRCUIT FOUND. ENTER CHECK.` **Meaning:** The command SELECT QUERY was entered and an unidentifiable circuit was found. **System Action:** None **User Action:** To remove the unidentifiable circuit, enter the CHECK command.
—end—

Usage Note:

When numbers 00 to 09 are entered as 0 to 9, the system default interprets the unentered first character as 0 and accepts the abbreviated entry. This applies to all BERP commands that require a number to be input.

Examples:

1 The following example shows how to SELECT an IBERT Line Card (ILC) and reserve it for the BERP test.

```
> SELECT ADD L HOST 0 1 8 8
```

2 The following example shows how to remove an IBERT DTU from the table.

```
> SELECT REMOVE DTU 0
```

3 The following example shows how to list all IBERTs reserved for use.

```
> SELECT QUERY
```

4 The following example shows how to select all IBERT DTUs in the switch for use.

```
> SELECT ALL DTU
```

5 The following example shows how to clear all selected IBERT Line Cards from the internal table.

> SELECT CLEAR ILC

```
         | ADD      TRK  pm_type pm_no  link_no  ckt_no
         | REMOVE  L     [site]      len       [loopbk] DGRP [dgrp_no]
         |               LCD              ff   u      [dd]
         |               LM                ff  u       [lnk]
 DEFINE  |               LINK pm_type pm_no [link_no]
         |
         | CLEAR
         | QUERY
```

DEFINE

This command is used to define trunks, data line cards, links, LSGs, and LM links.

where:

ADD prepares to add a trunk, data line card (DLC), LSG, or link to the pool for testing.
REMOVE prepares to remove a trunk, DLC, LSG, or link from the pool.
TRK indicates that a trunk is to be specified.
L indicates that a DLC is to be specified.
LCD is Line Concentrating Device which indicates that an LSG is to be specified.
LM is Line Module which indicates that an LM link is to be specified.
LINK indicates that a link is to be specified.
pm_type specifies the peripheral module (pm) type:

DTC	(digital trunk controller)
IAC	(ISDN Access Controller)
LTC	(line trunk controller)
LTCI	(line trunk controller ISDN)
RCC	(remote cluster controller)
DCM	(digital carrier module)

pm_no specifies the peripheral module number from 0 to 255.
link_no specifies the link number from 0 to 19. The default is all datafilled link numbers.
ckt_no specifies the circuit number from 1 to 31. The default is all datafilled circuit numbers.
site is the location of the DLC, LCD, or LM (such as HOST or

REM2 etc.).

len — is the LEN of the DLC, LCD, or LM. For DLC, it is in the form ff u dd cc; for LCD, in the form ff u dd; for LM, in the form ff u lnk.

where:

ff — is the frame number from 00 to 99.
u — is the unit number from 0 to 9.
dd — is the drawer number from 00 to 19. It is optional for LCD; if omitted, all drawers (LSGs) on that LCD will be added or removed.
cc — is the circuit number from 00 to 31.
lnk — is the LM link. The default value for it is all datafilled links (1 to 3) on the LM.

loopbk — indicates the type of loopback to be specified.

DLC — is the default entry for DLC
ISLC — is the entry for ISDN cards with a 2B1Q loop (i.e. NTBX27AA)
LSG — is the default entry for endpoints other than DLC or ISLC. For LCME lens, the entry expands to include DGRP and specifies the digroup number.

dgrp_no — values 0, 1 or 2 are valid digroup numbers
CLEAR — clears all the reserved LSGs, trunks, DLCs and links from the pool.
QUERY — displays a list of LSGs, trunks, DLCs and links defined for testing.

Responses

```
NO CIRCUITS/LINKS ARE DEFINED FOR TESTING.
```

Meaning: The DEFINE CLEAR command was entered when nothing was defined for testing.

System Action: None

User Action: None

```
NO CIRCUITS HAVE BEEN DEFINED FOR TESTING.
```

—continued—

Responses

Meaning: The DEFINE REMOVE command was entered when nothing was defined for testing.

System Action: None

User Action: None

```
THIS CIRCUIT HAS NOT BEEN DEFINED FOR TESTING.
```

Meaning: The command DEFINE REMOVE was entered for a circuit which was not defined for testing.

System Action: None

User Action: None

```
CIRCUIT REMOVED.
```

Meaning: The command DEFINE REMOVE was entered and the specified circuit was removed from testing.

System Action: None

User Action: None

```
CIRCUIT HAS BEEN RESERVED FOR LATER USE BY BERP.
```

Meaning: The command DEFINE ADD was entered for a circuit which was already defined for testing.

System Action: None

User Action: None

```
CIRCUIT IS NOT VALID FOR BERP TESTING.
```

Meaning: The command DEFINE ADD was entered for a circuit that can not be tested at the BERP level.

System Action: None

User Action: None

```
LINKS HAVE ALREADY BEEN DEFINED FOR TESTING.
```

—continued—

Responses

Meaning: The DEFINE ADD command was entered for a trunk or DLC while a link was defined for testing.

System Action: None

User Action: Links can not be defined in the same pool with trunks and DLCs. Enter the DEFINE CLEAR command before defining trunks or DLCs.

```
THIS PM/LINK HAS NOT BEEN DEFINED FOR TESTING.
```

Meaning: The DEFINE REMOVE command was entered for a trunk or link when when neither was specified for testing.

System Action: None

User Action: None

```
PM REMOVED.
```

Meaning: The DEFINE REMOVE command was entered for a trunk or link and the specified peripheral module was removed from testing.

System Action: None

User Action: None

```
LINK REMOVED.
```

Meaning: The DEFINE REMOVE command was entered for a link and the specified link was removed.

System Action: None

User Action: None

```
LINK WAS NOT DEFINED FOR TESTING.
```

Meaning: The command DEFINE REMOVE was entered for a link when no links were defined for testing.

System Action: None

User Action: None

```
NO MORE ROOM FOR ADDITIONAL PMS.
```

—continued—

Responses

Meaning: The command DEFINE ADD was entered for a trunk or link when the maximum number of peripheral modules (20) was already defined.

System Action: None

User Action: Use the DEFINE REMOVE command to make room for another peripheral module.

```
CIRCUITS HAVE ALREADY BEEN DEFINED FOR TESTING.
```

Meaning: The command DEFINE ADD was entered for a circuit which was already defined for testing.

System Action: None

User Action: None

```
THE FOLLOWING LINKS HAVE BEEN DEFINED:
```

Meaning: The command DEFINE ADD was entered for trunks or links and a list of links on the specified peripheral module was displayed.

System Action: None

User Action: None

```
NOTE: THE FOLLOWING EQUIPPED LINKS ARE NOT MANB OR HAVE A
LOOPBACK SET:
```

```
THESE LINKS CANNOT BE TESTED WITH THE CURRENT LOOPBACK
SETTING.
```

Meaning: The command DEFINE ADD was entered for trunks or links but some of the links on the peripheral module could not be tested because the link was not compatible with the specified loopback.

System Action: None

User Action: Change the loopback for the defined links.

```
NO OTHER VALID LINKS EXIST FOR TESTING.
```

—continued—

Responses

Meaning: The command DEFINE ADD was entered for a trunk or link but other than the listed links, no other valid links exist on the peripheral module.

System Action: None

User Action: None

```
NOTE: THE FOLLOWING LINKS ARE REMOTELY LOOPED BACK:
```

Meaning: The command DEFINE ADD was entered for a trunk or link but some of the links on the peripheral module can not be tested because they are remotely looped back.

System Action: None

User Action: Change the loopback for the defined links.

```
ERROR - PM IS NOT VALID FOR BERP TESTING
ERROR - PM IS NOT IN SERVICE
ERROR - LINKS NOT EQUIPPED
```

Meaning: The command DEFINE ADD was entered for a trunk or link but the peripheral module was not acceptable for the reason specified by the error message.

System Action: None

User Action: If the peripheral module is not in service, then RTS the peripheral module.

```
NO CIRCUIT DATAFILLED AT THAT LOCATION.
```

Meaning: The command DEFINE ADD was entered for a trunk circuit which was not datafilled.

System Action: None

User Action: Recheck the specified trunk circuit number.

```
LINK pm_type pm_no IS NOT ACCEPTABLE. REASON: PM INVALID
LINK pm_type pm_no IS NOT ACCEPTABLE. REASON: UNEQUIPPED
LINK pm_type pm_no IS NOT ACCEPTABLE. REASON: PM BUSY
LINK pm_type pm_no IS NOT ACCEPTABLE. REASON: UNAVAILABLE
```

—continued—

Responses

```
LINK pm_type pm_no IS NOT ACCEPTABLE. REASON: LOCAL
LOOPBACK IS NOT COMPATIBLE WITH CURRENT LOOPBK SETTING.
```

```
LINK pm_type pm_no IS NOT ACCEPTABLE. REASON: REMOTE
LOOPBACK IS NOT COMPATIBLE WITH CURRENT LOOPBK SETTING.
```

Meaning: The command DEFINE ADD was entered for a trunk or link but the link was not acceptable for one of the reasons given. The following is a list of possible reasons:

Unequipped - The link is not datafilled.

Unavailable - The link is not in a MANB state.

PM Busy - The PM is not in service.

Failed - Failed to set the loopback.

Local Loopback - Link has a local loopback set.

Remote Loopback- Link has a remote loopback set.

System Action: None

User Action: Check the status of the link to be tested.

```
NUMBER OF PMs WITH LINKS DEFINED:
```

```
THE FOLLOWING LINKS ON DTC 0 HAVE BEEN DEFINED:
```

```
TOTAL NUMBER OF LINKS DEFINED:
```

```
TOTAL NUMBER OF CCTS ON ON LINKS:
```

```
loopback_type LOOPBACK WILL BE USED FOR LINK TESTING.
```

Meaning: The command DEFINE QUERY was entered and the display shows the peripheral modules, links, circuits and loopbacks.

System Action: None

User Action: None

```
UNIDENTIFIABLE CIRCUIT FOUND. ENTER CHECK.
```

—continued—

Responses

Meaning: The command DEFINE QUERY was entered and an unidentifiable circuit was found (typically a deleted circuit).

System Action: None

User Action: Enter the CHECK command to remove the circuit.

```
n CIRCUITS HAVE BEEN SELECTED FOR THE IBERTS TO CONNECT TO
’list of line equipment numbers trunks and CLLIs’
```

Meaning: The DEFINE QUERY command was entered and the list of reserved DLCs and trunks is displayed. The n represents the total number reserved for use.

System Action: None

User Action: None

```
INVALID LINE EQUIPMENT NUMBER ENTERED.
```

Meaning: The DEFINE ADD or DEFINE REMOVE command was entered for a DLC and an invalid LEN was specified.

System Action: None

User Action: Check the LEN and re-enter the command.

```
UNKNOWN LOOP BACK ENTERED

CIRCUIT NOT ADDED
```

Meaning: The entered value for the LOOPBK parameter was other than LSG or DLC.

System Action: None

User Action: Enter the correct loopback value (LSG or DLC).

```
INVALID LOOPBACK FOR THIS LINE.

CIRCUIT NOT ADDED
```

—continued—

Responses

Meaning: User specified wrong type of LOOPBK for the type of line defined for testing.

System Action: None

User Action: Re-enter command after correcting the loopback value for the type of line defined for testing.

```
CIRCUIT IS NOT SUPPORTED BY BERP. CIRCUIT NOT ADDED
```

Meaning: The entered type of line for BERP testing is not supported by BERP.

System Action: None

User Action: Re-enter command with the correct line type (POTS, EBS, or DATA) for BERP testing.

```
INVALID PM ENTERED, PM NOT ADDED FOR TESTING
```

Meaning: The PM entered by the user is not an LCM or LCMI.

System Action: None

User Action: Re-enter command with the PM type of LCM or LCMI.

```
LSG OUT OF RANGE. PM NOT ADDED FOR TESTING
```

Meaning: The specified LSG number is out of range.

System Action: None

User Action: Specify the within range (0 to 19) LSG number and re-enter the command.

```
YOU HAVE ALREADY DEFINED LINKS TO BE TESTED.

YOU MUST ISSUE A DEFINE CLEAR BEFORE ADDING LSG FOR TESTING
```

—continued—

Responses

Meaning: The test is already defined for links and the user tried to add LSGs. The LSGs must be tested separately from the links.

System Action: None

User Action: Either clear the links with DEFINE CLEAR command and then add the LSGs with DEFINE ADD command, or continue with the links test.

```
YOU HAVE ALREADY DEFINED LMs TO BE TESTED.
```

```
YOU MUST ISSUE A DEFINE CLEAR BEFORE ADDING LSG FOR TESTING
```

Meaning: The test is already defined for LMs and the user tried to add LSGs. The LSGs must be tested separately from the LMs.

System Action: None

User Action: Either clear the LMs with DEFINE CLEAR command and then add the LSGs with DEFINE ADD command, or continue with the LM test.

```
THE FOLLOWING LSGs WILL BE TESTED: 0, 1, 2, 3
```

Meaning: User defined an LCM or LCMI without specifying any LSG. The system defaults to all equipped LSGs on the given LCM or LCMI and includes them in the test.

System Action: None

User Action: None

```
NO MORE ROOM TO STORE THIS PM FOR AN LSG TEST
```

Meaning: User defined an LCM or LCMI specifying an LSG when the maximum allowed (20) had already been defined.

System Action: Outputs the message

User Action: Either remove some of the LSGs and add the desired LSG, or start the test.

```
LSG HAS BEEN REMOVED FROM TESTING.
```

—continued—

Responses

Meaning: LSG specified for removal by the user was removed from test

System Action: None

User Action: None

```
LAST LSG ON THIS PM REMOVED FROM TESTING.
PM HAS BEEN COMPLETELY REMOVED FROM TESTING
```

Meaning: Since the user specified the removal of the last LSG of a PM defined for testing, the PM itself was removed from test.

System Action: None

User Action: None

```
PM HAS BEEN REMOVED FROM TESTING.
```

Meaning: PM specified for removal by the user was removed from test

System Action: None

User Action: None

```
PM NOT FOUND, NO ENTRY REMOVED FROM TESTING
```

Meaning: PM specified for removal by the user had not been defined for BERP testing.

System Action: None

User Action: Use the DEFINE QUERY command to see which PMs are defined for BERP testing.

```
LSG n IS UNEQUIPPED, NOT ADDED FOR TESTING
```

Meaning: LSG n, which the user specified for testing, is in the unequipped state and, as such, not valid for testing.

System Action: None

User Action: Define an LSG which is not in the unequipped state.

```
NO EQUIPPED LSGs ON THIS LCD TO TEST. LCD NOT ADDED
```

—continued—

Responses

Meaning: User specified an LCD which has no LSG in the equipped state and, therefore, is not valid for testing.

System Action: None

User Action: Define an LCD which has at least one LSG in the equipped state.

```
YOU HAVE ALREADY DEFINED LINKs TO BE TESTED.
YOU MUST ISSUE A DEFINE CLEAR BEFORE ADDING LMs FOR TESTING
```

Meaning: User tried to add an LM to a test which is already defined for links. It is not allowed, because links must be tested separately.

System Action: None

User Action: Either use DEFINE CLEAR command to clear the links and then define LMs, or continue with the link test.

```
YOU HAVE ALREADY DEFINED LSGs TO BE TESTED.
YOU MUST ISSUE A DEFINE CLEAR BEFORE ADDING LMs FOR TESTING
```

Meaning: User tried to add an LM to a test which is already defined for LSGs. It is not allowed, because LSGs must be tested separately.

System Action: None

User Action: Either use DEFINE CLEAR command to clear the LSGs and then define LMs, or continue with the LSG test.

```
YOU HAVE ALREADY DEFINED CIRCUITS TO BE TESTED.
YOU MUST ISSUE A DEFINE CLEAR BEFORE ADDING LMs FOR TESTING
```

Meaning: User tried to add an LM to a test which is already defined for circuits. It is not allowed, because circuits must be tested separately.

System Action: None

User Action: Either use DEFINE CLEAR command to clear the circuits and then define LMs, or continue with the circuit test.

```
INPUT PM IS NOT AN LM
```

—continued—

Responses

Meaning: When defining an LM for testing, the user specified the PM information that was not that of an LM. Therefore the PM was not added for testing.

System Action: None

User Action: Re-enter the command with the valid LM information.

```
RLMs ARE NOT SUPPORTED BY BERP
```

Meaning: User specified an RLM instead of an LM for testing. RLM is not supported and therefor was not added for testing.

System Action: None

User Action: Re-enter the command with the valid LM information.

```
NO MORE ROOM TO STORE THIS PM FOR AN LM TEST
```

Meaning: User specified an LM for testing when the maximum allowed (20) had already been defined. New entry was not added for testing.

System Action: None

User Action: Either remove some of the entries and then add new LM, or start the test.

```
INPUT LM LINK WILL BE TESTED
```

Meaning: User specified an LM link for testing; it was added to the test.

System Action: None

User Action: None

```
ALL LINKS ON THE LM WILL BE TESTED
```

—continued—

Responses

Meaning: User specified an LM for testing without the link parameter. All datafilled links (1 to 3) on the LM will be tested.

System Action: None

User Action: None

```
LINK n IS OUT OF RANGE, NOT ADDED FOR TESTING
```

Meaning: User specified a link which was not datafilled for the input LM.

System Action: None

User Action: Re-enter the command specifying a datafilled link on the LM for the test.

```
LINK WAS NOT DEFINED TO BE TESTED
```

Meaning: User specified a link, which had not been defined for test, to be removed from the test.

System Action: None

User Action: Use DEFINE QUERY command to check which links have been defined for the test.

```
LAST LINK ON LM REMOVED FROM TESTING
LM HAS BEEN COMPLETELY REMOVED FROM TESTING
```

Meaning: User specified a link for removal from the test and it was the last link on the LM to be tested. Therefore, the LM was also removed from the test.

System Action: None

User Action: None

```
LINK REMOVED FROM TESTING
```

Meaning: User specified link was removed from the test.

System Action: None

User Action: None

```
LM HAS BEEN COMPLETELY REMOVED FROM TESTING
```

—continued—

Responses	
	Meaning: User specified an LM for removal from the test, without specifying any link. The LM was removed from the test. **System Action:** None **User Action:** None
`LM NOT FOUND, NO ENTRY REMOVED FROM TESTING`	**Meaning:** User specified an LM for removal from the test, but the LM had not been defined for test. No entry was removed. **System Action:** None **User Action:** Use DEFINE QUERY command to check which LMs are defined for testing.
—end—	

Usage Note:

The numbers 00 to 09 can be entered as 0 to 9.

Examples:

1 The following example shows how to DEFINE a DLC for testing.

> DEFINE ADD L HOST 0 1 8 8

2 The following example shows how to DEFINE an LSG for testing.

> DEFINE ADD LCD 0 1 5

3 The following example shows how to DEFINE an LM link for testing.

> DEFINE ADD LM 0 0 2

4 The following example shows how to define the digroup of the LCME for testing. This parameter is valid for LCME lens

> DEFINE ADD L HOST 3 1 0 26 LSG DGRP 1

5 The following example shows how to define an ISDN line card. This parameter is valid for line cards such as NTBX27AA with a 2B1Q line format.

> DEFINE ADD L HOST 3 1 0 26 ISLC

6 The following example shows how to remove a trunk from the pool.

> DEFINE REMOVE DTC 0 0 1

7 The following example shows how to define a link.

> DEFINE ADD LINK DTC 0

8 The following example shows how to list all circuits defined for use.

> DEFINE QUERY

9 The following example shows how to clear all DEFINEd circuits.

> DEFINE CLEAR

CALLSET	LENGTH	x	HOURS MINS
	DELAY	y	HOURS MINS SECS
	CALLS	z CONT	
	ALLCALLS ERRORED		

CALLSET

This command is used to set up the call parameters for a BERP test. These parameters include the length of each test call, the length of the delay between test calls, the number of calls to be made, and the calls to be traced by BERP.

where:

LENGTH	prepares to establish the duration of a test call.
x	specifies the number of units for the length of the test call from 1 to 255.
DELAY	prepares to establish the duration of the delay between calls.
y	specifies the number of units for the delay between test calls

from 0 to 255.
HOURS specifies the units for x and/or y.
MINS specifies the units for x and/or y.
SECS specifies the units for y.
CALLS prepares to establish the number of calls to be made during the test.
z specifies the number of calls from 1 to 32767.
CONT specifies that the test calls continue to be made until a STOP command is issued.
ALLCALLS specifies that all calls are to be traced by BERP.
ERRORED specifies that only the errored calls are to be traced by BERP.

Responses

```
THE CALL LENGTH FOR EACH TEST CALL IS X (MINS OR HOURS).
```

Meaning: The parameter LENGTH was specified to set the duration of the test calls between 1 and 255 (mins or hours).

System Action: None

User Action: None

```
THE DELAY BETWEEN EACH TEST CALL WILL BE Y (HOURS, MINS OR
SECS).
```

Meaning: The parameter DELAY was specified to set the duration of the delay between test calls from 0 to 255 (hours, mins or secs).

System Action: None

User Action: None

```
A MINIMUM OF Z CALLS WILL BE MADE.
```

Meaning: The parameter CALLS was specified to set the number of calls between 0 and 32767.

System Action: None

User Action: None

```
ALL CALLS WILL BE TRACED.
```

—continued—

Responses	
	Meaning: The parameter ALLCALLS was specified to trace all the calls by BERP. **System Action:** None **User Action:** None
`ERRORED CALLS WILL BE TRACED.`	
	Meaning: The parameter ERRORED was specified to trace only the errored calls by BERP. **System Action:** None **User Action:** None
—end—	

Usage Note:

The default setting is ERRORED calls. At reload restart, the parameter is set to this value. The cold and warm restarts donot affect the current setting (ERRORED or ALLCALLS).

Example:

1 The following examples show how to set the call data.

```
> CALLSET LENGTH 1 MINS DELAY 2 MINS CALLS 50
> CALLSET LENGTH 2 MINS DELAY 3 MINS CALLS CONT
> CALLSET LENGTH 3 HOURS DELAY 1 MINS
> CALLSET LENGTH 1 MINS CALLS 40
> CALLSET LENGTH 30 MINS
> CALLSET DELAY 1 MINS CALLS 50
> CALLSET DELAY 2 MINS
> CALLSET CALLS 60
> CALLSET ALLCALLS
```

DEFTIME	START	SET day hh mm
	STOP	[CLEAR]
	CLEAR	

DEFTIME

This command is used to set the start and/or stop time of the test. It can also be used to clear the start and/or stop times.

where:

START prepares to establish when to start the test.
STOP prepares to establish when to stop the test.
SET prepares to establish the day and time for starting or stopping the test.
day establishes the day a test will start or stop. Either MON, TUE, WED, THU, FRI, SAT or SUN.
hh establishes the hour that the test will start or stop, where hh is the hour from 0 to 23.
mm establishes the minutes that the test will start or stop, where mm is the minutes from 00 to 59.
CLEAR as an optional subparameter of the START parameter, clears the start setting. As a subparameter of the STOP parameter, clears the stop setting.
CLEAR as a parameter of the DEFTIME command, clears both the start and stop settings.

Responses

```
START TIME HAS TO BE EARLIER THAN THE STOP TIME.
```

Meaning: The start time that was entered was later than the previously defined stop time.

System Action: None

User Action: Re-enter a start time that is earlier than the stop time.

```
START TIME IS NOW SET AT : YY/MM/DD HH MM SS DAY
```

Meaning: The command to set the start time was entered and this message confirms the start information.

System Action: None

User Action: None

```
START TIME HAS BEEN CLEARED.
```

—continued—

Responses

Meaning: The command DEFTIME START CLEAR was entered and this confirms that the information was cleared.

System Action: None

User Action: None

```
STOP TIME HAS TO BE LATER THAN THE START TIME.
```

Meaning: The command to set the stop time was entered and the specified stop time was earlier than the previously specified start time.

System Action: None

User Action: Re-enter a stop time that is later than the start time.

```
STOP TIME IS NOW SET AT : YY/MM/DD HH MM SS DAY
```

Meaning: The command to set the stop time was entered and this message confirms the information.

System Action: None

User Action: None

```
STOP TIME HAS BEEN CLEARED.
```

Meaning: The command DEFTIME STOP CLEAR was entered and this confirms that the stop information was cleared.

System Action: None

User Action: None

```
START TIME HAS BEEN CLEARED.
STOP TIME HAS BEEN CLEARED.
```

Meaning: The command DEFTIME CLEAR was entered and this confirms that both the start and stop times were cleared.

System Action: None

User Action: None

—end—

Usage Note:

Either a ’stop time’ or a ’number of calls’ can be specified, but not both.

Examples:

1 The following example shows how to set a start time.

```
> DEFTIME START SET MON 11 00
```

2 The following example shows how to clear a start time.

```
> DEFTIME START CLEAR
```

3 The following example shows how to set a stop time.

```
> DEFTIME STOP SET TUE 7 30
```

4 The following example shows how to clear a stop time.

```
> DEFTIME STOP CLEAR
```

5 The following example shows how to clear both the start and stop time.

```
> DEFTIME CLEAR
```

CHECK	

CHECK

This command checks all test data for consistency and validity. If all data is consistent then the test status is set to CHECKED.

When the CHECK command is entered, the following actions are performed:

- Checks that the start time has not already passed. If a start time has not been specified, the tests begin as soon as a start command is issued.
- Checks that the start and stop times have both been defined and verify that the stop time is later than the start time.
- Check that the stop time (if defined) has not already passed. If a stop time has not been defined and the number of calls to make is set to 0, the tests continue until a stop command is issued.
- Checks that IBERT cards have been specified and that they are properly datafilled.
- Checks that the number of endpoints to be tested is 0, or that the number of endpoints is equal to or greater than the number of IBERTs. The endpoints are also checked to ensure that they are properly datafilled.
- Checks that either a stop time or a number of calls made is entered but not both.
- Checks that an output file has been specified.

- Removes unidentifiable circuits from the internal table.

Responses

```
ERROR - UNIDENTIFIABLE DEFINED CIRCUIT FOUND.
```

Meaning: The CHECK command was entered. An unidentifiable circuit was found and removed from the internal table.

System Action: None

User Action: None

```
ERROR - UNIDENTIFIABLE SELECTED IBERT FOUND.
```

Meaning: The CHECK command was entered. An unidentifiable circuit was found and removed from the internal table.

System Action: None

User Action: None

```
UNIDENTIFIABLE IBERTS HAVE BEEN REMOVED.
```

Meaning: The CHECK command was entered and all unidentifiable IBERTs were removed.

System Action: None

User Action: None

```
UNIDENTIFIABLE CIRCUITS HAVE BEEN REMOVED.
```

Meaning: The CHECK command was entered and all unidentifiable circuits were removed.

System Action: None

User Action: None

```
ERROR - INVALID SELECTED IBERT FOUND:
```

—continued—

Responses

Meaning: The CHECK command was entered and an invalid IBERT was found.

System Action: None

User Action: Enter the SELECT REMOVE command to remove this IBERT.

```
ERROR - INVALID DEFINED CIRCUIT FOUND:
```

Meaning: The CHECK command was entered and an invalid circuit was found.

System Action: None

User Action: Enter the DEFINE REMOVE command to remove this circuit.

```
ERROR - ONLY n VALID CIRCUITS EXIST ON LINKS DEFINED. THERE
MUST BE AT LEAST AS MANY IBERTS SELECTED AS CIRCUITS TO
TEST.
```

Meaning: The CHECK command was entered and there are more IBERTs than there are circuits to test.

System Action: None

User Action: Either DEFINE more circuits for testing or SELECT REMOVE IBERTs from testing.

```
ERROR - LINK pm_type pm_no IS NOT ACCEPTABLE. REASON: PM
INVALID
```

```
ERROR - LINK pm_type pm_no IS NOT ACCEPTABLE. REASON:
UNEQUIPPED
```

```
ERROR - LINK pm_type pm_no IS NOT ACCEPTABLE. REASON: PM
BUSY
```

```
ERROR - LINK pm_type pm_no IS NOT ACCEPTABLE. REASON:
UNAVAILABLE
```

```
ERROR - LINK pm_type pm_no IS NOT ACCEPTABLE. REASON: LOCAL
LOOPBACK IS NOT COMPATIBLE WITH CURRENT LOOPBK SETTING.
```

```
ERROR - LINK pm_type pm_no IS NOT ACCEPTABLE. REASON:
REMOTE LOOPBACK IS NOT COMPATIBLE WITH CURRENT LOOPBK
SETTING.
```

—continued—

Responses

Meaning: The CHECK command was entered and the link was not acceptable for one of the reasons given. The following is a list of possible reasons:

Unequipped - The link is not datafilled.

Unavailable - The link is not in a MANB state.

PM Busy - The PM is not in service.

Failed - Failed to set the loopback.

Local Loopback - Link has a local loopback set.

Remote Loopback - Link has a remote loopback set.

System Action: None

User Action: Verify the status of the link to be tested.

```
WARNING - NO OUTPUT FILE HAS BEEN DEFINED.
```

Meaning: The CHECK command indicated that there was no output file defined.

System Action: None

User Action: If an output file is desired, issue the OUTPUT DEFINE command to define an output file.

```
OUTPUT FILE NAME EXISTS ON THE OUTPUT DEVICE.

PLEASE SPECIFY A UNIQUE OUTPUT FILE NAME OR A DIFFERENT
OUTPUT DEVICE.
```

Meaning: The CHECK command indicated that an output file with the same name exists on the output device. The output file name must be unique to the file device.

System Action: None

User Action: Issue the OUTPUT CLEAR command followed by the OUTPUT DEFINE command.

```
ERROR - STOP TIME HAS ALREADY PASSED.
```

—continued—

Responses

Meaning: The CHECK command indicated that the stop time had already passed.

System Action: None

User Action: Issue the DEFTIME STOP CLEAR command to clear the existing stop time, then issue the DEFTIME STOP SET command to set a new stop time.

```
ERROR - STOP TIME MUST BE LATER THAN START TIME.
```

Meaning: The CHECK command indicated that the stop time was not later than the start time.

System Action: None

User Action: Issue the DEFTIME STOP CLEAR command to clear the existing stop time, then issue the DEFTIME STOP SET command to set a new stop time.

```
ERROR - INCOMPATIBLE STOP CRITERIA. CANNOT DEFINE BOTH A
STOP TIME AND NUMBER OF CALLS TO BE MADE.
```

Meaning: The CHECK command indicated that both a stop time and a number of calls had been specified.

System Action: None

User Action: Either set the calls to make to 0 by issuing the command CALLSET CALLS 0, or clear the stop time by issuing the DEFTIME STOP CLEAR command.

```
ERROR - START TIME HAS ALREADY PASSED.
```

Meaning: The CHECK command indicated that the start time had already passed.

System Action: None

User Action: Issue the DEFTIME START CLEAR command to clear the start time, then issue the DEFTIME START SET command to define a new start time.

```
ERROR - START TIME MUST BE EARLIER THAN STOP TIME.
```

—continued—

Responses

Meaning: The CHECK command indicated that the start time was not earlier than the stop time.

System Action: None

User Action: Issue the DEFTIME START CLEAR command to clear the start time, then issue the DEFTIME START SET command to define a new start time.

```
NO START TIME HAS BEEN DEFINED, TESTS WILL BEGIN
IMMEDIATELY AFTER THE START COMMAND.
```

Meaning: The CHECK command indicated that a start time had not been specified.

System Action: None

User Action: Either use the START command to start the test manually, or define a start time with the DEFTIME START SET command.

```
ERROR - YOU MUST DATAFILL AT LEAST AS MANY CIRCUITS AS YOU
HAVE SELECTED IBERT CARDS.
```

Meaning: There are not enough circuits for the IBERTs to connect to.

System Action: None

User Action: Issue the DEFINE command to specify more circuits.

```
ERROR - YOU HAVE NOT DEFINED ANY IBERTS FOR USE.
```

Meaning: The CHECK command indicated that IBERT cards had not been selected.

System Action: None

User Action: Issue the SELECT command to specify more IBERT cards for use.

```
WARNING - SUSPECT OR FAULTY IBERT FOUND.
```

—continued—

Responses

Meaning: There is a suspect or faulty IBERT among the IBERTs selected for use in BERP testing.

System Action: None

User Action: Run a diagnostic on the IBERT. If it passes the diagnostic, it is functional; if it fails, it is faulty and should be removed by using the command SELECT REMOVE.

```
ERROR - BERP DOES NOT HAVE ACCESS TO THIS IBERT
```

Meaning: The specified IBERT cannot be used by BERP because of datafill in tables FMRESINV and FMRESUSE.

System Action: None

User Action: Check datafill or remove the IBERT using the command SELECT REMOVE.

```
ERROR - TOO MANY CIRCUITS ON PM, MAXIMUM n ALLOWED
```

Meaning: The PM has more circuits defined on it than the number of IBERTs that can connect to them.

System Action: None

User Action: Delete some of the circuits on the PM or add more IBERTs.

```
ERROR - EXTRANEOUS LINE DEFINED FOR TESTING, MUST BE
REMOVED
```

Meaning: A line was defined for testing, but its associated PM was not defined.

System Action: None

User Action: Remove the line or add the associated PM information.

```
ERROR - NO LINES DEFINED ON PM
```

Meaning: No lines were defined on the PM for the IBERTs to connect to for an LSG test.

System Action: None

User Action: Define a line on the PM or remove the PM from the test.

—continued—

Responses

```
THE FOLLOWING CIRCUIT(S) MUST BE REMOVED USING THE DEFINE
REMOVE COMMAND IN ORDER TO DO AN LSG TEST: CIRCUITS IDs
```

Meaning: All circuits must be lines for the LSG test, but circuits other than lines were defined.

System Action: None

User Action: Remove these circuits by using the DEFINE REMOVE command.

```
UNKNOWN PM FOUND, ENTRY BEING DELETED
```

Meaning: A PM was deleted from the list of PMs to be tested because it could not be identified.

System Action: None

User Action: Re-enter CHECK command after the deletion.

```
NON LCM TYPE NODE FOUND IN TABLE, DELETED
```

Meaning: A PM was deleted from the list of PMs to be tested after its data was changed to a type other than an LCM.

System Action: None

User Action: Re-enter CHECK command after the deletion.

```
NO LOOPBK FOUND FOR CIRCUIT ID
```

Meaning: DEFAULT loopback was specified for the circuit, but the CHECK command was unable to determine the default loopback for that type of circuit.

System Action: None

User Action: Remove the circuit and re-enter CHECK command.

```
LOOPBK SETTING NOT VALID FOR CIRCUIT ID
```

Meaning: The type of loopback set is not valid for the identified circuit.

System Action: None

User Action: Either set a different (correct) type of loopback by using the LOOPBK command, or remove the circuit.

—continued—

Responses

```
INVALID LOOPBACK SETTING FOR CIRCUIT TEST
```

Meaning: The type of loopback set is not valid for a circuit test.

System Action: None

User Action: Set the loopback to be one of LSG, DLC, or DEFAULT by using the LOOPBK command.

```
LOOPBACK SETTING IS NOT VALID FOR THIS TYPE OF TESTING
```

Meaning: The type of loopback set was one of LSG, DLC, or DEFAULT when a set of non-LM links was defined for testing.

System Action: None

User Action: Set the loopback to be one of DS1, DS30, or NONE by using the LOOPBK command.

```
LOOPBACK SETTING WILL BE IGNORED FOR LSG TEST
```

Meaning: For LSG testing, any loopback setting other than DEFAULT is ignored.

System Action: None

User Action: None

```
LOOPBACK SETTING WILL BE IGNORED FOR LM TEST
```

Meaning: For LM testing, any loopback setting other than DEFAULT is ignored.

System Action: None

User Action: None

```
ALL DATA VERIFIED, TESTS MAY NOW BE STARTED.
```

Meaning: The CHECK command indicated that all of the data was consistent and valid. The test status was set to CHECKED.

System Action: None

User Action: Issue the START command.

—end—

Usage Note:

No special characteristics apply to this command.

Example:

1 The following example shows how to enter the CHECK command.

> CHECK

or

> 8

START	

START

This command issues a request to start the tests at the specified start time. The same validity and consistency checks are performed for the START command as for the CHECK command. This may result in the same error messages as the CHECK command error messages.

Responses

```
UNABLE TO START THE TEST. (MAILBOX RETURN CODE)
```

Meaning: A problem was encountered in trying to start the test.

System Action: None

User Action: Contact system support with the mailbox return code.

```
TESTS HAVE BEEN STARTED.
```

Meaning: The START command was entered and the test was started immediately.

System Action: None

User Action: None

```
UNABLE TO SET THE START TIME FOR THE TEST. (MAILBOX RETURN
CODE)
```

—continued—

Responses

Meaning: An attempt was made to set the start up for a later time and a problem was encountered during the process.

System Action: None

User Action: Contact system support with the mailbox return code.

```
UNEXPECTED MESSAGE TYPE RECEIVED. (MT = MESSAGE TYPE)
```

Meaning: The START command was entered and an attempt was made to set-up the start at a later time. A problem was encountered during the process and the message type was output in the response message.

System Action: None

User Action: Contact system support with the message type that was output in response message.

```
START TIME HAS BEEN SET.
```

Meaning: The START command was entered and a start time was set for a later time. No problems were encountered.

System Action: None

User Action: None

```
A TEST IS WAITING TO BEGIN, YOU MUST ISSUE THE STOP COMMAND
FIRST.
```

Meaning: The START command was entered on a test which was already waiting to start.

System Action: None

User Action: Allow the previously scheduled tests to run, or issue the STOP command to clear the start time.

```
TESTS ARE ALREADY RUNNING, YOU MUST ISSUE THE STOP COMMAND
FIRST.
```

—continued—

Responses
Meaning: The START command was entered on a test which was already running. **System Action:** None **User Action:** Allow the running test to finish, or issue the STOP command to stop the test.
`TESTS ARE BEING STOPPED, WAIT UNTIL THEY HAVE FULLY STOPPED BEFORE STARTING ANOTHER.`
Meaning: The START command was entered on a test which was in the process of stopping. **System Action:** None **User Action:** Allow the tests to fully stop.
—end—

Usage Note:

No special characteristics apply to this command.

Example:

1 The following example shows how to enter the START command.

> START

or

> 10

STOP	

STOP

This command immediately attempts to stop a currently running test. Any active test calls are stopped and are not considered as part of the performance summary.

Responses

```
UNABLE TO STOP THE TEST (RC = MAILBOX RETURN CODE)
```

Meaning: The command STOP was entered and the attempt was unsuccessful.

System Action: None

User Action: Try the command again. If the same message appears contact system support with the mailbox return code.

```
UNEXPECTED MESSAGE TYPE RECEIVED. (MT = MESSAGE TYPE)
```

Meaning: The command STOP was entered and a problem was encountered during the process.

System Action: None

User Action: Try the command again. If the same message appears contact system support with the message type.

```
TESTS STOPPED.
```

Meaning: The STOP command was entered and the test was stopped immediately.

System Action: None

User Action: None

```
TESTS ARE IN THE PROCESS OF STOPPING, BE PATIENT.
```

Meaning: The command STOP was entered on a test which was in the process of stopping.

System Action: None

User Action: None

—continued—

Responses
`NO TEST IS CURRENTLY RUNNING.` **Meaning:** The STOP command was entered when there was no test running. **System Action:** None **User Action:** None
—end—

Usage Note:

No special characteristics apply to this command.

Example:

1 The following example shows how to enter the STOP command.

> STOP

or

> 11

OUTPUT	DEFINE filename filedevice CLEAR

OUTPUT

This command is used to specify the output file, or to clear the previous definition of the output file.

where:

DEFINE	prepares to define the name of the output file and the output device.
filename	is the filename (such as BERPFILE)
filedevice	is the output device name (such as SFDEV)
CLEAR	clears the specified file name and file device.

Responses

```
ERROR - AN OUTPUT FILE HAS ALREADY BEEN CREATED, YOU MUST
ISSUE THE OUTPUT CLEAR COMMAND.
```

Meaning: The OUTPUT DEFINE command was entered while the previous output file was open.

System Action: None

User Action: Issue the OUTPUT CLEAR command to clear the file name and file device.

```
OUTPUT FILE NAME ALREADY EXISTS ON THE OUTPUT DEVICE.
PLEASE SPECIFY A UNIQUE OUTPUT FILE NAME OR A DIFFERENT
OUTPUT FILE DEVICE.
```

Meaning: The OUTPUT DEFINE command was entered and the specified output file name already existed on the device.

System Action: None

User Action: Issue the OUTPUT CLEAR command to clear the specified file name and device, then issue the OUTPUT DEFINE command to specify a different file name or file device.

```
OUTPUT FILE HAS BEEN REMEMBERED, IT WILL BE CREATED WHEN
THE TESTS ARE STARTED.
```

Meaning: The command OUTPUT DEFINE was entered successfully.

System Action: None

User Action: None

```
OUTPUT FILE INFORMATION HAS BEEN CLEARED.
```

Meaning: The command OUTPUT CLEAR was entered and the previous information was erased.

System Action: None

User Action: None

```
COULD NOT CLOSE PREVIOUS FILE.
```

—continued—

Responses
Meaning: The OUTPUT CLEAR command was entered and the previous file could not be closed. **System Action:** None **User Action:** Contact system support.
`PROBLEM GETTING OUTPUT VOLUME INFORMATION.` **Meaning:** The OUTPUT DEFINE command was entered and the specified device could not be found. **System Action:** None **User Action:** Check the device specified and re-issue the command.
—end—

Usage Note:

No special characteristics apply to this command.

Examples:

1 The following example shows how to define a file named BERPFILE and a device named SFDEV.

> OUTPUT DEFINE BERPFILE SFDEV

2 The following example shows how to clear the file name and device.

> OUTPUT CLEAR

RESET	[ALL]

RESET

This command resets all statistical counters to 0. The following are the counts that are reset:

- Number of calls made.
- Number of error free calls.
- Number of errored calls.
- Number of call set-up failures.
- Number of failures to seize lines.

- Number of no sync found calls.
- Number of calls with a Bit Error Ratio worse than the value specified by the user.
- Number of calls with more errored seconds than the value specified by the user.

Note: This command may be issued while tests are running. Any counts up to that point are lost.

where:

ALL prompts for a yes or no response before resetting the test set-up data: IBERTs, BER exponent, DLCs and errored seconds, etc.

Responses

```
RESET
```

Meaning: The statistical counters have been reset.

System Action: None

User Action: None

```
WARNING, ALL TEST SET-UP DATA WILL BE RESET.  PLEASE
CONFIRM (”YES” OR ”NO”):
```

Meaning: Requires a response to be entered at the MAP.

System Action: None

User Action: Enter YES to reset the test set-up data, or NO to leave the set-up data as is.

Usage Note:

RESET ALL is not valid on a running or waiting test.

Example:

1 The following example shows how to enter the RESET command.

```
> RESET
```

or

> 12

PARMSET	BER ber_exponent SECONDS errored_seconds

PARMSET

This command is used to set the Bit Error Ratio exponent parameter and the error free seconds parameter of the test.

where:

BER prepares to establish the Bit Error Ratio exponent
exponent establishes the value of the exponent from 1 to 9.
SECONDS prepares to establish the number of errored seconds from 0 to 9.
errored_seconds establishes the number of seconds.

Responses

```
The Bit Error Ratio (BER) and the errored seconds will
appear on the MAP display as entered.
```

Meaning: The BERP display is updated with the BER and the number of errored seconds.

System Action: None

User Action: None

Usage Note:

No special characteristics apply to this command.

Example:

1 The following examples show how to change the BER and/or the number of errored seconds. The BERP display changes to reflect the new exponent and the number of errored seconds.

```
> PARMSET BER 6 SECONDS 2
> PARMSET BER 7
> PARMSET SECONDS 2
```

SORTKEY	ADDKEY DELKEY	nodetype	key
	ADDNODE DELNODE	nodetype	
	ALL CLEAR		
	QUERY	[nodetype]	

SORTKEY

This command is used to determine the path components to be included in the sorting of path information by the PROCESS command. It can include or remove a specified node and/or sortkey (component) for processing.

where:

ADDKEY specifies the node and sortkey which determine the path components to be included for processing.

DELKEY removes the specified node and sortkey for processing.

ADDNODE specifies the node (with default sortkey) to be included for processing.

DELNODE removes the specified node for processing.

ALL Includes all nodetypes for processing, with their respective default sortkeys

CLEAR removes all nodetypes for processing.

QUERY queries sortkeys to be used for all nodetypes, or for a specific node.

nodetype is the type of a node used in a BERP test call. The various nodetypes are: NET TM LTC LM LCM DCM RCC IAC

where:

NET indicates the nodetype is network.

TM indicates the nodetype is Trunk Module. It also includes MTM

LTC indicates the nodetype is Line Trunk Controller. It also includes LGC DTC LTCI LGCI

LM indicates the nodetype is Line Module.

LCM indicates the nodetype is Line Concentrating Module. It also includes LCMI.

DCM indicates the nodetype is Digital Carrier Module.

RCC indicates the nodetype is Remote Cluster Controller.

IAC indicates the nodetype is ISDN Access Controller.

key is the sortkey that determines the path components to be included for processing. The sortkeys are specified according to the associated nodetype.

The various sortkeys are: U P CH CL CLCH PL PLCH LS LSCH JR JRCH LK LKCH XP XPCH PCL UCL UPL ULS

where:

U signifies an active unit and is an LCM sortkey.

P signifies plane and is one of the sortkeys of a TM.

CH signifies channel and is one of the sortkeys of a TM.

CL signifies Cside link and is one of the sortkeys of LTC, LM, LCM, DCM, RCC, and IAC.

CLCH signifies Cside link and channel. It is one of the sortkeys of LTC, LCM, DCM, RCC, and IAC.

PL signifies Pside link and it is one of the sortkeys of LTC, DCM, RCC, and IAC.

PLCH signifies Pside link and channel. It is one of the sortkeys of LTC, DCM, RCC, and IAC.

LS signifies line subgroup. It is one of the sortkeys of LCM.

LSCH signifies line subgroup and channel. It is one of the sortkeys of LCM.

JR signifies junctor. It is one of the sortkeys of NET.

JRCH signifies junctor and channel. It is one of the sortkeys of NET.

LK signifies link and is one of the sortkeys of NET.

LKCH signifies link and channel. It is one of the sortkeys of NET.

XP signifies cross point and is one of the sortkeys of NET.

XPCH signifies cross point and channel. It is one of the sortkeys of NET.

PCL signifies plane and Cside link. It is one of the sortkeys of LTC, LM, DCM, and IAC.

UCL signifies unit and Cside link. It is one of the sortkeys of LTC, LCM, RCC, and IAC.

UPL signifies unit and Pside link. It is one of the sortkeys of

LTC, RCC, and IAC.

ULS signifies unit and line subgroup. It is one of the sortkeys of LCM.

Responses

```
DEFAULT SORTKEYS ADDED FOR ALL NODETYPES.
```

Meaning: Default sortkeys for all nodetypes are included for BERP processing.

System Action: None

User Action: None

```
SORTKEYS REMOVED FOR ALL NODETYPES.
```

Meaning: All nodetypes are excluded for BERP processing.

System Action: None

User Action: None

```
SORTKEY ADDED.
```

Meaning: The specified sortkey for a specified nodetype is included for BERP processing.

System Action: None

User Action: None

```
NODETYPE HAS NO SORTKEYS SPECIFIED.
```

Meaning: The specified nodetype had no sortkeys specified.

System Action: None

User Action: None

```
DEFAULT SORTKEYS ADDED FOR NODETYPE.
```

Meaning: The default sortkeys for the specified nodetype are added when the SORTKEY command is entered with ADDNODE parameter.

System Action: None

User Action: None

```
SORTKEYS REMOVED FOR NODETYPE.
```

—continued—

Responses

Meaning: The specified nodetype is excluded for BERP processing when the SORTKEY command is entered with DELNODE parameter.

System Action: None

User Action: None

```
SORTKEY HAS ALREADY BEEN ADDED.
```

Meaning: The specified sortkey has been previously added.

System Action: None

User Action: None

```
SORTKEY WAS NOT PREVIOUSLY ADDED.
```

Meaning: The specified sortkey for deletion had not been previously added.

System Action: None

User Action: None

```
DEFAULT SORTKEYS ALREADY SPECIFIED.
```

Meaning: The default sortkeys already specified when the SORTKEY command with ADDNODE parameter is entered.

System Action: None

User Action: None

```
CURRENT SORTKEYS REPLACED WITH DEFAULT.
```

Meaning: The currently specified sortkeys have been replaced with the default sortkeys for the specified nodetype.

System Action: None

User Action: None

```
NODETYPE: SORTKEY ... SORTKEY.
```

—continued—

Responses	
	Meaning: When queried, the sortkeys for the specified nodetype are listed. **System Action:** None **User Action:** None
`NODETYPE: NO SORTKEYS SPECIFIED.`	**Meaning:** The sortkeys for the specified nodetype have not been specified. **System Action:** None **User Action:** None
—end—	

Usage Note:

The specified sortkeys are not affected by the cold and warm restarts, but they are cleared by the reload restart.

Example:

1 The following examples show how to enter typical SORTKEY commands.

> SORTKEY ADDKEY LCM CL
> SORTKEY DELKEY LCM CL
> SORTKEY ADDNODE LCM
> SORTKEY DELNODE LCM
> SORTKEY ALL
> SORTKEY QUERY

PROCESS	string	ERRORED FORMAT WORST (0 TO 100) CARDS (0 TO 100)

PROCESS

This command is used to process the Bit Error Rate Performance result file and produce a report file based on the BERP input file data.

where:

string specifies the inname, indevice, outname, outdevice.

where:

inname specifies the name of the input file.
indevice specifies the input file device (volume).
outname specifies the name of the output file.
outdevice specifies the output file device.

ERRORED specifies that if the input file contains non-errored paths, these paths should be ignored and only the errored paths be processed.

FORMAT specifies that, in addition to summary and sorted path information, a complete path trace of each errored call is output.

WORST specifies the number of worst components (with an errored call ratio worse than 1/1000) to be output. The range is from 0 to 100. If the WORST parameter is not specified, the default is 20.

CARDS specifies the maximum number of worst cards (determined by the ratio of errored calls to total calls made through a particular card) to be output. The range is from 0 to 100. If the CARDS parameter is not specified, the default is 20. If less than the specified number of cards experienced errors, only those cards that experienced errors are output. If a value of 0 is specified, or if only errored paths are traced during the test (the ERRORED parameter is specified) then a report for worst cards is not generated.

Responses

```
OUTPUT FILE NAME ALREADY EXISTS, PLEASE RE-ISSUE THE
COMMAND AND SPECIFY A FILE NAME UNIQUE TO THE OUTPUT
DEVICE.
```

Meaning: An output file was specified to send the processed results to and the output file already existed.

System Action: None

User Action: Re-issue the command specifying an output file which does not already exist on the output device.

```
INPUT FILE IS NOT A BERP RESULT FILE.
```

—continued—

Responses

Meaning: An input file was specified which was not a BERP result file.

System Action: None

User Action: Re-issue the command specifying an input file which is a BERP result file.

```
COULD NOT OPEN THE INPUT FILE, REASON IS FILE RETURN CODE.
```

Meaning: A system error occurred while trying to open the input file. The error is described by the file return code.

System Action: None

User Action: None

```
COULD NOT CREATE THE OUTPUT FILE, REASON IS FILE RETURN
CODE.
```

Meaning: A system error occurred while trying to create the output file. The error is described by the file return code.

System Action: None

User Action: None

```
PROBLEM GETTING THE VOLUME INFORMATION.
```

Meaning: The specified device (disk or tape drive) for either the input or output file was not a known device.

System Action: None

User Action: Check the device name and re-issue the command.

```
INPUT FILE DOES NOT EXIST.
```

Meaning: The specified input file did not exist.

System Action: None

User Action: Check the name specified and re-issue the command.

```
READING IN THE INPUT FILE...
```

—continued—

Responses

Meaning: The process command is reading in the input file and storing the path information.

System Action: None

User Action: None

```
NUMBER OF PATHS RECORDS READ IN IS : n
```

Meaning: The process command read in n path records from BERP input file.

System Action: None

User Action: None

```
ALLOCATED STORE FOR x CALLS AND y NODES.
```

Meaning: Store has been allocated for maximum number of x calls and y nodes.

System Action: None

User Action: None

```
CALLS TO SORT: x NODES TO SORT: y.
```

Meaning: The number of calls and nodes to be processed is x and y respectively.

System Action: None

User Action: None

```
GENERATING THE OUTPUT REPORT.
```

Meaning: The process command is generating the output report.

System Action: None

User Action: None

```
NO CALLS TO PROCESS.
```

—continued—

Responses

Meaning: After reading in the BERP input file and including the nodes specified by the SORTKEY command, there are no calls left to process.

System Action: None

User Action: None

```
WARNING - NO SORTKEYS HAVE BEEN SPECIFIED. NO SORTED PATH
INFORMATION WILL BE GENERATED. DO YOU WISH TO CONTINUE?
PLEASE CONFIRM (”YES” or ”NO”)
```

Meaning: No sortkeys were specified with the SORTKEY command when the PROCESS command was entered.

System Action: None

User Action: Enter ”NO” and then use the SORTKEY command to specify sortkeys. Otherwise, enter ”YES” and no sorted path information will be generated.

```
INPUT FILE CONTAINS NON-ERRORED CALLS. THESE CALLS WILL BE
IGNORED.
```

Meaning: The PROCESS command specified ERRORED option to process only the errored calls. The input file contains non-errored calls, which will be ignored.

System Action: None

User Action: None

```
INPUT FILE DOES NOT CONTAIN NON-ERRORED CALLS. ”ERRORED”
OPTION IGNORED.
```

Meaning: The PROCESS command specified ERRORED option to process only the errored calls, but the input file does not contain non-errored calls. Therefore, the ERRORED option is ignored.

System Action: None

User Action: None

```
SORTING NODES... FINISHED SORTING NODES.
```

—continued—

Responses

Meaning: This indicates that the BERP process is sorting the input file components.

System Action: None

User Action: None

```
PROCESSING TIME: hh mm ss.mmm
```

Meaning: This is the amount of time required to process the input file.

System Action: None

User Action: None

```
INSUFFICIENT STORE TO SAVE ALL CALLS. PROCESSING SAVED
CALLS...
```

Meaning: The input file contains more calls/nodes than store could be allocated to handle. The currently saved calls/nodes are processed first; the remaining calls in the input file are processed as a second batch.

System Action: None

User Action: None

```
ALL DONE.
```

Meaning: The process command finished generating the output report.

System Action: None

User Action: None

```
FAILED TO ALLOCATE STORE FOR SORTING.
```

Meaning: No store could be allocated for processing the input file.

System Action: None

User Action: Contact system support.

```
ABORTING DUE TO OUTPUT FILE PROBLEM, REASON : FILE RETURN
CODE.
```

—continued—

Responses

Meaning: A system error occurred while writing to the output file. The reason is given by the file return code.

System Action: None

User Action: Contact system support.

```
ERROR ON THE INPUT FILE, REASON IS : FILE RETURN CODE.
```

Meaning: A system error occurred while reading from the input file. The reason is given by the file return code.

System Action: None

User Action: Contact system support.

—end—

Usage Note:

No special characteristics apply to this command.

Example:

1 The following examples show how to enter typical PROCESS commands.

> PROCESS BERPFILE SFDEV RESULTS SFDEV FORMAT
> PROCESS BERPFILE SFDEV RESULTS SFDEV ERRORED
> PROCESS BERPFILE SFDEV RESULTS SFDEV WORST
> PROCESS BERPFILE SFDEV RESULTS SFDEV

LOOPBK	QUERY DS30 DS1 LSG DLC DEFAULT NONE

LOOPBK

This command is used to select the loopback to be used for BERP testing.

where:

QUERY displays the current loopback setting.

DS30	specifies that a DS30 loopback will be used for link testing.
DS1	specifies that a DS1 loopback will be used for link testing.
LSG	specifies that an LSG loopback will be used for lines supported for BERP testing.
DLC	specifies that a DLC loopback will be used for data lines.
DEFAULT	specifies that a DEFAULT loopback will be used for testing.
NONE	specifies that no loopback will be used for testing.

Responses

```
NO LOOPBACK WILL BE USED FOR LINK TESTING. DS1 LOOPBACK
WILL BE USED FOR LINK TESTING. DS30 LOOPBACK WILL BE USED
FOR LINK TESTING.
```

Meaning: The command LOOPBK QUERY was entered and one of the loopback messages is displayed.

System Action: None

User Action: None

```
ERROR - COMMAND IS NOT VALID AT THIS POINT. YOU MUST ISSUE
THE STOP COMMAND FIRST.
```

```
ERROR - TESTS ARE RUNNING. YOU MUST ISSUE THE STOP COMMAND
FIRST.
```

```
ERROR - TESTS ARE STOPPING. PLEASE WAIT UNTIL THEY HAVE
COMPLETELY STOPPED.
```

Meaning: The command LOOPBK was entered and the loopback could not be set for the reason given.

System Action: None

User Action: Take the appropriate action indicated by the error message.

```
LSG LOOPBACK WILL BE USED FOR TESTING.
```

Meaning: User entered LOOPBK command with parameter LSG to select LSG loopback for the test.

System Action: None

User Action: None

```
DLC LOOPBACK WILL BE USED FOR TESTING.
```

—continued—

Responses

Meaning: User entered LOOPBK command with parameter DLC to select DLC loopback for the test.

System Action: None

User Action: None

```
DEFAULT LOOPBACK WILL BE USED FOR TESTING.
```

Meaning: User entered LOOPBK command with parameter DEFAULT so that the default loopback for the given circuit is selected.

System Action: None

User Action: None

```
INVALID LOOP BACK FOR THE TYPE OF TEST DEFINED
```

Meaning: The type of loopback specified by the user is not supported for the type of test which has been defined.

System Action: None

User Action: Re-enter command with the correct loop- back parameter.

```
INVALID LOOP BACK FOR LINK TEST
```

Meaning: User specified one of DLC, LSG, or DEFAULT as loopback when a link test had been defined.

System Action: None

User Action: Re-enter command with the correct loop- back (DS1, DS30, or NONE) for link test.

—end—

Usage Note:

If LOOPBK NONE is selected, it is assumed that an external loopback has been established for the DS1 link.

Examples:

1 The following example shows how to set the loopback to DS30.

> LOOPBK DS30

2 The following example shows how to set the loopback to LSG.

> LOOPBK LSG

3 The following example shows how to query the selected loopback.

> LOOPBK QUERY

General error messages

These general error messages appear when any command that modifies the test data is entered while the test status is any one of UNCHECKED, CHECKED OR STOPPED.

Responses
`ERROR - THIS IS NOT VALID AT THIS POINT. YOU MUST ISSUE THE STOP COMMAND FIRST.` **Meaning:** This general error message is generated for all commands that try to alter any of the test data if the status is WAITING. **System Action:** None **User Action:** None
`ERROR - TESTS ARE RUNNING, YOU MUST ISSUE THE STOP COMMAND FIRST.` **Meaning:** This general error message is generated for all commands that try to alter any of the test data if the status is RUNNING. **System Action:** None **User Action:** None
`ERROR - TESTS ARE STOPPING, PLEASE WAIT UNTIL THEY HAVE COMPLETELY STOPPED.` **Meaning:** This general error message is generated for all commands that try to alter any of the test data if the status is STOPPING. **System Action:** None **User Action:** None

List of terms

BER	Bit Error Ratio
BERP	Bit Error Rate Performance
BERT	Bit Error Rate Test
CHNL	Channel
DCM	Digital Carrier Module
DLC	Data Line Card
DMS	Digital Multiplex System
DTC	Digital Trunk Controller
DTU	Digital Test Unit
IAC	ISDN Access Controller
IBERT	Integrated Bit Error Rate Tester

ICTS

Integrity Check Traffic Simulator

ILC

IBERT Line Card

JCT

Junctor

LCD

Line Concentrating Device

LCM

Line Concentrating Module

LCMI

Line Concentrating Module ISDN

LGC

Line Group Controller

LGCI

Line Group Controller ISDN

LM

Line Module

LNK

Link

LSG

Line Subgroup

LTC

Line Trunk Controller

LTCI

Line Trunk Controller ISDN

MAP

Maintenance and Administration Position

MTM

Maintenance Trunk Module

NET

Network

NET PATH

Network Path Diagnostics

PM

Peripheral Module

RCC

Remote Cluster Controller

TM

Trunk Module

XBERT

XPM Bit Error Rate Tester

XPM

XMS–based Peripheral Module

XPT

Crosspoint

DMS-100 Family

Bit Error Rate Performance Testing

Testing Reference

NORTHERN TELECOM CONFIDENTIAL: The information contained in this document is the property of Northern Telecom. Except as specifically authorized in writing by Northern Telecom, the holder of this document shall keep the information contained herein confidential and shall protect same in whole or in part from disclosure and dissemination to third parties and use same for evaluation, operation, and maintenance purposes only.

Information is subject to change without notice. Northern Telecom reserves the right to make changes in design or components as progress in engineering and manufacturing may warrant.This equipment has been tested and found to comply with the limits for a Class A digital device pursuant to Part 15 of the FCC Rules, and the radio interference regulations of the Canadian Department of Communications. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference in which case the user will be required to correct the interference at the user’s own expense

Allowing this equipment to be operated in such a manner as to not provide for proper answer supervision is a violation of Part 68 of FCC Rules, Docket No. 89-114, 55FR46066

The SL-100 system is certified by the Canadian Standards Association (CSA) with the Nationally Recognized Testing Laboratory (NRTL).

This equipment is capable of providing users with access to interstate providers of operator services through the use of equal access codes. Modifications by aggregators to alter these capabilities is a violation of the Telephone Operator Consumer Service Improvement Act of 1990 and Part 68 of the FCC Rules

Publication number: 297-1001-533
Product release: TELECOM10
Document release: Standard 07.01
Date: April 1999
Printed in the United States of America

NORTEL
NORTHERN TELECOM