

**Agilent Technologies 8960 Series 10 E5515A,B Wireless Communications Test Set  
Agilent Technologies E1960A GSM Mobile Test Application**

# **Programming Guide**

Test Application Revision A.04

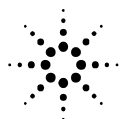
© Copyright Agilent Technologies 1998, 1999

Printed in U.S.A. March 2000

Agilent Part Number: E1960-90002

**Revision G**

**<http://www.agilent.com/find/8960support/>**



**Agilent Technologies**

---

## Notice

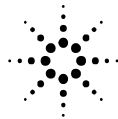
Information contained in this document is subject to change without notice.

All Rights Reserved. Reproduction, adaptation, or translation without prior written permission is prohibited, except as allowed under the copyright laws.

This material may be reproduced by or for the U.S. Government pursuant to the Copyright License under the clause at DFARS 52.227-7013 (APR 1988).

Agilent Technologies, Inc.  
Learning Products Department  
24001 E. Mission  
Liberty Lake, WA 99019-9599

U.S.A.



**Agilent Technologies**

---

# **Safety, Warranty, and Regional Sales and Service Offices Information**

### **Manufacturer's Declaration**

This statement is provided to comply with the requirements of the German Sound Emission Directive, from 18 January 1991.

This product has a sound pressure emission (at the operator position) < 70 dB(A).

- Sound Pressure  $L_p < 70$  dB(A).
- At Operator Position.
- Normal Operation.
- According to ISO 7779:1988/EN 27779:1991 (Type Test).

### **Herstellerbescheinigung**

- Schalldruckpegel  $L_p < 70$  dB(A).
- Diese Information steht im Zusammenhang mit den Anforderungen der Maschinenlärminformationsverordnung vom 18 Januar 1991.
- Am Arbeitsplatz.
- Normaler Betrieb.
- Nach ISO 7779:1988/EN 27779:1991 (Typprüfung).

## Safety Considerations

### GENERAL

This product and related documentation must be reviewed for familiarization with safety markings and instructions before operation.


This product has been designed and tested in accordance with *IEC Publication 1010*, "Safety Requirements for Electronic Measuring Apparatus," and has been supplied in a safe condition. This instruction documentation contains information and warnings which must be followed by the user to ensure safe operation and to maintain the product in a safe condition.


### SAFETY EARTH GROUND

A uninterruptible safety earth ground must be provided from the main power source to the product input wiring terminals, power cord, or supplied power cord set.

### SAFETY SYMBOLS

 Indicates instrument damage can occur if indicated operating limits are exceeded.

 Indicates hazardous voltages.

 Indicates earth (ground) terminal

---

**WARNING**     **A WARNING note denotes a hazard. It calls attention to a procedure, practice, or the like, which, if not correctly performed or adhered to, could result in personal injury. Do not proceed beyond a WARNING sign until the indicated conditions are fully understood and met.**

---

---

**CAUTION**     A CAUTION note denotes a hazard. It calls attention to an operation procedure, practice, or the like, which, if not correctly performed or adhered to, could result in damage to or destruction of part or all of the product. Do not proceed beyond an CAUTION note until the indicated conditions are fully understood and met.

---

**WARNING**

**This product is a Safety Class I instrument (provided with a protective earthing ground incorporated in the power cord). The mains plug shall only be inserted in a socket outlet provided with a protective earth contact. Any interruption of the protective conductor inside or outside of the product is likely to make the product dangerous. Intentional interruption is prohibited.**

**Whenever it is likely that the protection has been impaired, the instrument must be made inoperative and be secured against any unintended operation.**

**If this instrument is to be energized via an autotransformer (for voltage reduction), make sure the common terminal is connected to the earth terminal of the power source.**

**If this product is not used as specified, the protection provided by the equipment could be impaired. This product must be used in a normal condition (in which all means for protection are intact) only.**

**No operator serviceable parts in this product. Refer servicing to qualified personnel. To prevent electrical shock, do not remove covers.**

**Servicing instructions are for use by qualified personnel only. To avoid electrical shock, do not perform any servicing unless you are qualified to do so.**

**The opening of covers or removal of parts is likely to expose dangerous voltages. Disconnect the product from all voltage sources while it is being opened.**

**The power cord is connected to internal capacitors that may remain live for 5 seconds after disconnecting the plug from its power supply.**

**For Continued protection against fire hazard, replace the line fuse(s) only with 250 V fuse(s) or the same current rating and type (for example, normal blow or time delay). Do not use repaired fuses or short circuited fuseholders.**

**Always use the three-prong ac power cord supplied with this product. Failure to ensure adequate earth grounding by not using this cord may cause product damage.**

**This product is designed for use in Installation Category II and Pollution Degree 2 per *IEC 1010* and *IEC 664* respectively. FOR INDOOR USE ONLY.**

**This product has autoranging line voltage input, be sure the supply voltage is within the specified range.**

**To prevent electrical shock, disconnect instrument from mains (line) before cleaning. Use a dry cloth or one slightly dampened with water to clean the external case parts. Do not attempt to clean internally.**

**Ventilation Requirements: When installing the product in a cabinet, the convection into and out of the product must not be restricted. The ambient temperature (outside the cabinet) must be less than the maximum operating temperature of the product by 4° C for every 100 watts dissipated in the cabinet. If the total power dissipated in the cabinet is greater than 800 watts, then forced convection must be used.**

---

## Product Markings

CE - the CE mark is a registered trademark of the European Community. A CE mark accompanied by a year indicated the year the design was proven.

CSA - the CSA mark is a registered trademark of the Canadian Standards Association.

## CERTIFICATION

*Agilent Technologies, Inc. certifies that this product met its published specifications at the time of shipment from the factory. Agilent Technologies further certifies that its calibration measurements are traceable to the United States National Institute of Standards and Technology, to the extent allowed by the Institute's calibration facility, and to the calibration facilities of other International Standards Organization members*

## WARRANTY

This Agilent Technologies instrument product is warranted against defects in material and workmanship for a period of one year from date of shipment. During the warranty period, Agilent Technologies, Inc. will at its option, either repair or replace products which prove to be defective.

For warranty service or repair, this product must be returned to a service facility designated by Agilent. Buyer shall prepay shipping charges to Agilent and Agilent shall pay shipping charges, duties, and taxes for products returned to Agilent from another country.

Agilent warrants that its software and firmware designated by Agilent for use with an instrument will execute its programming instructions when properly installed on that instrument. Agilent does not warrant that the operation of the instrument, or software, or firmware will be uninterrupted or error free.

## LIMITATION OF WARRANTY

The foregoing warranty shall not apply to defects resulting from improper or inadequate maintenance by Buyer, Buyer-supplied software or interfacing, unauthorized modification or misuse, operation outside of the environmental specifications for the product, or improper site preparation or maintenance.

NO OTHER WARRANTY IS EXPRESSED OR IMPLIED. AGILENT SPECIFICALLY DISCLAIMS THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

## EXCLUSIVE REMEDIES

THE REMEDIES PROVIDED HEREIN ARE BUYER'S SOLE AND EXCLUSIVE REMEDIES. AGILENT SHALL NOT BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, WHETHER BASE ON CONTRACT, TORT, OR ANY OTHER LEGAL THEORY.

## ASSISTANCE

*Product maintenance agreements and other customer assistance agreements are available for Agilent Technologies products. For any assistance, contact your nearest Agilent Technologies Sales and Service Office.*

## DECLARATION OF CONFORMITY

according to ISO/IEC Guide 22 and EN 45014

Manufacturer's Name: **Agilent Technologies Inc.**  
Manufacturer's Address: **24001 E. Mission Avenue  
Liberty Lake, Washington 99019-9599  
USA**

declares that the product

Product Name: Agilent Technologies 8960 Series 10  
Wireless Communications Test Set  
Model Number: Agilent Technologies E5515A,B  
Product Options: This declaration covers all options of  
the above product.

conforms to the following Product specifications:

Safety: IEC 1010-1:1990+A1+A2 / EN 61010-1:1993

EMC: CISPR 11:1990/EN 55011:1991- Group 1, Class A  
EN 50082-1 : 1992  
IEC 801-2:1991 - 4kV CD,8kV AD  
IEC 801-3:1984 3V/m  
IEC 801-4:1988 0.5 kV Sig. Lines, 1 kV Power Lines

### Supplementary Information:

This product herewith complies with the requirements of the Low Voltage Directive 73/23/EEC and the EMC Directive 89/336/EEC and carries the CE-marking accordingly.

Spokane, Washington USA November 20, 1998

  
Vince Roland  
Reliability & Regulatory  
Engineering Manager

European Contact: Your local Agilent Technologies Sales and Service Office or Agilent Technologies GmbH  
Department ZQ/Standards Europe, Herrenberger Strasse 130, D-71034 Böblingen, Germany (FAX+49-7031-14-3143)



**Table 1. Regional Sales and Service Offices**

<p>United States of America: Agilent Technologies Test and Measurement Call Center P.O. Box 4026 Englewood, CO 80155-4026</p> <p>(tel) 1 800 452 4844</p>	<p>Canada: Agilent Technologies Canada Inc. 5159 Spectrum Way Mississauga, Ontario L4W 5G1</p> <p>(tel) 1 877 894 4414</p>	<p>Europe: Agilent Technologies European Marketing Organization P.O. Box 999 1180 AZ Amstelveen The Netherlands</p> <p>(tel) (3120) 547 9999</p>
<p>Japan: Agilent Technologies Japan Ltd. Measurement Assistance Center 9-1 Takakura-Cho, Hachioji-Shi, Tokyo 192-8510, Japan</p> <p>(tel) (81) 456-56-7832 (fax) (81) 426-56-7840</p>	<p>Latin America: Agilent Technologies Latin America Region Headquarters 5200 Blue Lagoon Drive, Suite #950 Miami, Florida 33126 U.S. A.</p> <p>(tel) (305) 267 4245 (fax) (305) 267 4286</p>	<p>Australia/New Zealand: Agilent Technologies Australia Pty Ltd 347 Burwood Highway Forest Hill, Wictoria 3131</p> <p>(tel) 1 800 629 485 (Australia) (fax) (61 3) 9272 0749 (tel) 0 800 738 378 (New Zealand) (fax) (64 4) 802 6881</p>
<p>Asia Pacific: Agilent Technologies 19/F, Cityplaza One, 111 Kings Road, Taikoo shing, Hong Kong, SAR</p> <p>(tel) (852) 2599 7899 (fax) (852) 2506 9233</p>		



---

# Contents

## Introduction

Conventions Used in This Programming Guide .....	14
Purpose of This Programming Guide .....	14
How This Programming Guide Is Organized .....	14
How to Use This Programming Guide .....	16
About the Programming Examples Presented in This Programming Guide .....	16

## Step 1: Set the Test Set's Operating Mode to Active Cell

Background .....	17
Overview of Active Cell Operating Mode .....	17
Setting the Test Set's Operating Mode to Active Cell .....	18

## Step 2: Configure the Base Station Emulator (BSE)

Background .....	19
Configuring the Broadcast Channel Parameters .....	20
Configuring the Traffic Channel Parameters .....	22
Things That Can Go Wrong .....	23

## Step 3: Configure the Measurement Execution Parameters

Background .....	24
Overview .....	25
Configuring Measurement Averaging Parameters .....	26
Configuring Measurement Triggering Parameters .....	27
Configuring the Burst Synchronization Parameter .....	28
Configuring Measurement Timeout Parameters .....	29
Configuring Measurement Specific Parameters .....	30

## Step 4: Establish an Active Link with Mobile Station

Background .....	32
Overview .....	36
Process for Making a Base Station Originated Call .....	36
Process for Making a Mobile Station Originated Call .....	39

## Step 5: Set the Mobile Station's Operating Conditions

Overview .....	42
----------------	----

## Step 6: Make Measurements

Background .....	43
Things That Can Go Wrong .....	46

## Step 6a: Start Set Of Concurrent Measurements

Starting Measurements .....	47
-----------------------------	----

---

## Contents

Step 6b: Determine if a Measurement Is Done	
Background .....	49
Overview .....	49
Step 6c: Obtain a Set of Measurement Results	
Background .....	51
Overview .....	52
Step 7: Perform an Intra-Cell Handover	
Background .....	53
Performing an Intra-Cell Handover .....	53
Performing a Dual-Band Handover .....	54
Step 8: Disconnect the Mobile Station from the BSE	
Background .....	57
Overview .....	58
Terminating an Active Call from the Base Station Emulator .....	58
Terminating an Active Call from the Mobile Station .....	59
Comprehensive Program Example	
Example Program With Comments .....	64
Example Program Without Comments .....	69

# **Programming the Agilent Technologies 8960 Series 10 for GSM Mobile Testing in Active Cell Operating Mode**

## Introduction

### Conventions Used in This Programming Guide

Throughout this Programming Guide the term “test set” refers to an Agilent Technologies 8960 Series 10 wireless communications test set with the E1960A GSM mobile test application installed.

### Purpose of This Programming Guide

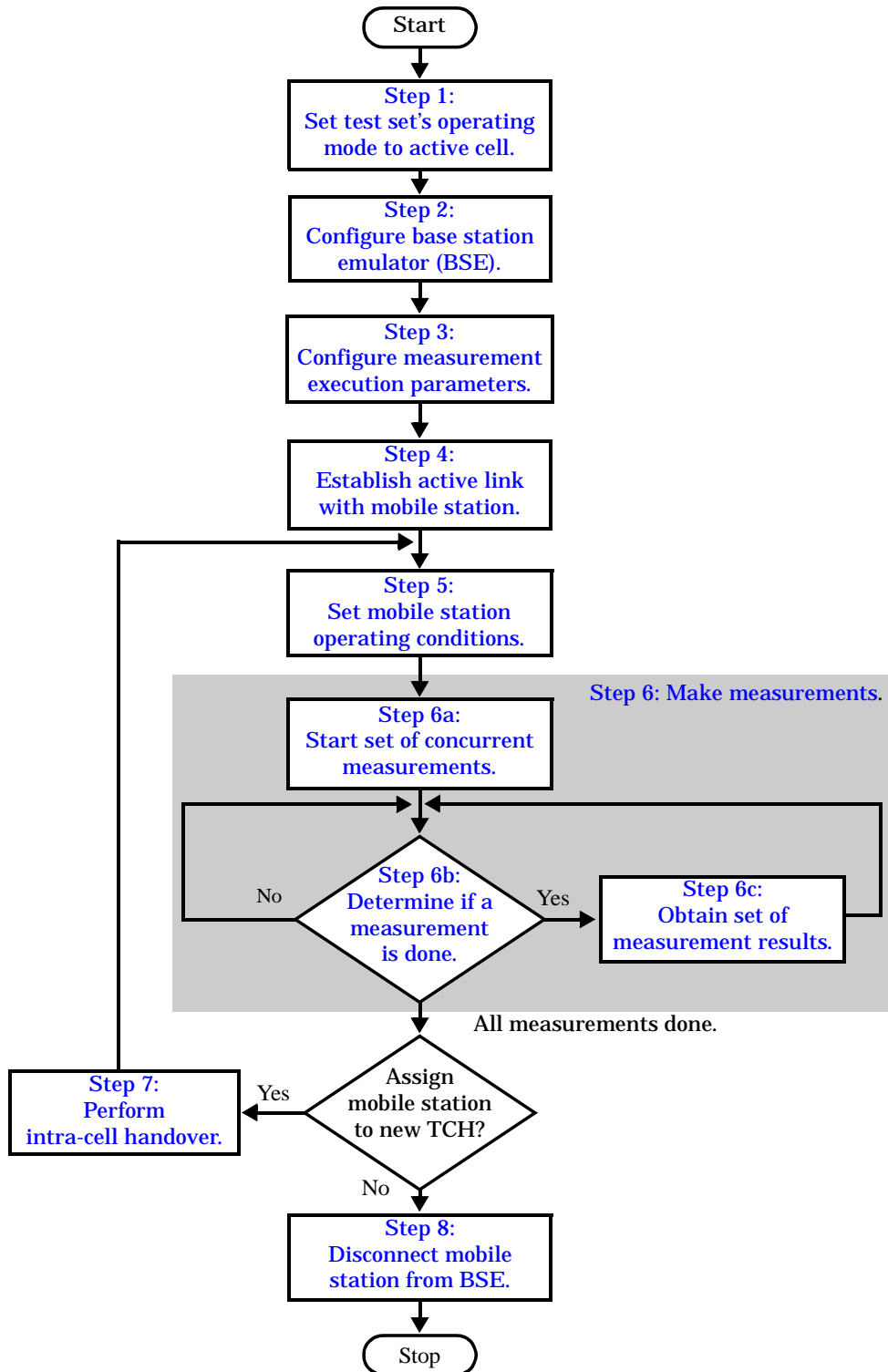
The test set represents state-of-the-art technology in one-box-testers and contains many powerful test capabilities which are accessible through easy-to-use GPIB programming commands. The purpose of this Programming Guide is to teach you how to write a basic control program, using the test set’s GPIB command set. This program will perform fundamental manufacturing tests on a GSM mobile station with the test set operating in active cell mode.

### How This Programming Guide Is Organized

The Programming Guide is organized around a typical set of tasks a control program would normally perform when testing a GSM mobile station in a manufacturing environment. The set of tasks is shown in “[Figure 1. Typical Flow of Tasks Performed by Control Program](#)” on page 15.

Typically in a manufacturing environment, steps 1, 2, and 3 are done once each time a production run is started, steps 4 and 8 are done once for each mobile station tested during the production run, and steps 5, 6, and 7 are done iteratively for each mobile station tested during the production run. The number of iterations for steps 5, 6, and 7 is dependent upon how many mobile station operating conditions are being tested (that is, number of channels, number of power levels, and so fourth).

Figure 1. Typical Flow of Tasks Performed by Control Program



## How to Use This Programming Guide

This Programming Guide is divided into 9 sections. Sections 1 through 8 (step 1 through 8) should be read in sequence. Each section, in order, discusses one of the tasks to be performed by the control program, showing how to accomplish that task using the test set's GPIB command set. As you progress through each section your understanding of how the test set's GPIB interface operates will increase as you see the control program evolve.

The last section of the Programming Guide presents a [“Comprehensive Program Example” on page 63](#) which uses all of the topics discussed in sections 1 through 8 together in one program to give the programmer a sense of how to tie everything together.

## About the Programming Examples Presented in This Programming Guide

### Programming Language:

Programming examples presented in this Programming Guide are written in the Rocky Mountain BASIC programming language, also known as RMB.

### Syntax Used in Programming Examples:

1. Programming examples use the shortened form of the command syntax to minimize GPIB bus transactions. The shortened form of a command is defined by use of capital letters in the command syntax.

#### Example 1. Command Syntax:

```
CALL:STATus:TCHannel:TSLot?
```

#### Example 2. Shortened Form:

```
CALL:STAT:TCH:TSL?
```

2. Programming examples do not include default nodes. Default nodes in the command syntax are defined by enclosing the node inside the [ ] brackets.

#### Example 3. Command Syntax:

```
CALL[:CELL[1]]:ACTivated[:STATe]<ON|1|OFF|0>
```

#### Example 4. Command Syntax without Default Nodes:

```
CALL:ACT <ON|1|OFF|0>
```

3. Programming examples make extensive use of compound commands using the ; and the ;; separators. Refer to the test set's reference information for information on the definition and use of these command separators.



## Step 1: Set the Test Set's Operating Mode to Active Cell

### Background

The test set contains a GSM base station emulator (BSE). The BSE's primary purpose is to provide the GSM call processing necessary for parametric measurements on the RF and audio signals of a GSM mobile station (MS).

An important characteristic of the test set's BSE is its operating mode. The operating mode sets the way in which the BSE interacts with the mobile station. The BSE has two operating modes; active cell mode and test mode.

Active cell mode is used when emulating a normal GSM cell. Test mode is used when it is not possible, or not desired, to communicate with the MS via over-the-air signaling, but downlink stimulus and uplink measurements are still needed.

This Programming Guide focuses on programming the test set's BSE in active cell operating mode.

### Overview of Active Cell Operating Mode

Active cell is the default operating mode of the test set's BSE and is used when emulating a normal GSM cell (that is, active signaling between the MS and the BSE).

### Active Cell Features

The basic features provided by the BSE when the operating mode is set to active cell are:

- Generation of a BCH (broadcast channel) without TCH (traffic channel).
- Support for location updating.
- Call setup, both MS and BSE originated.
- Changing TCH parameters during a call using over-the-air signaling.
- BSE initiated and MS initiated call disconnection.
- All measurements supported in the test application are available.
- The BSE automatically controls the test set's demodulation receiver.

Step 1: Set the Test Set's Operating Mode to Active Cell

## Setting the Test Set's Operating Mode to Active Cell

The test set's operating mode is set using the CALL:OPERating:MODE command.

### Example 1. Command Syntax:

```
CALL:OPERating:MODE <CELL|TEST>
```

### Example 2. Programming Example:

```
!*****  
! Step 1: Set Test Set Operating Mode To Active Cell  
!*****  
!  
OUTPUT Test_set;"CALL:OPER:MODE CELL
```

---

## Step 2: Configure the Base Station Emulator (BSE)

### Background

The test set contains a GSM base station emulator (BSE). In active cell operating mode the BSE, using the test set's GMSK modulated source, generates a downlink (BSE to MS direction) broadcast channel (BCH) which represents a cell. The MS can camp to this signal, just as it would camp to a cell on a real network. The BSE can then page the MS on the BCH and listen to the response of the MS on the uplink (MS to BSE direction), using the test set's demodulating receiver. Calls can then be set up with the establishment of a traffic channel (TCH) in both the downlink and uplink directions. Measurements can be made, using the BSE's measuring receiver, under essentially identical conditions to that which the MS would experience on a real network.

The BS Emulator can emulate a cell in any one of the following GSM frequency bands:

- PGSM - Primary (band) GSM, also known as GSM900
- EGSM - Extension (band) GSM (includes PGSM)
- DCS - Also known as DCS1800
- PCS - Also known as PCS1900

---

**NOTE** The term GSM is used to refer to any combination of, or all of, the supported bands. It is not used as a shortened term for PGSM.

---

The task of configuring the BSE consists of configuring the BCH and the TCH. There are numerous parameters that can be configured for both the BCH and the TCH. It may not be necessary to configure all the parameters all the time. The test set's default settings should allow a properly functioning MS to successfully camp on the cell under most circumstances.

In a manufacturing environment it may be desirable to explicitly configure the BCH and TCH parameters to ensure that the settings have not been corrupted by someone setting a parameter's value through the test set's front panel.

## Configuring the Broadcast Channel Parameters

The broadcast channel parameters are configured using the CALL processing subsystem commands shown in the following table.

**Table 1. Broadcast Channel Settable Parameters**

Parameter	Command Syntax	Footnote
Broadcast Band	CALL[:CELL[1]]:BAND <PGSM   EGSM   DCS   PCS>	1
Cell Power	CALL[:CELL[1]]:POWer[:AMPlitude]<numeric value>[<suffix>]	
Cell Power State	CALL[:CELL[1]]:POWer:STATe <ON   1   OFF   0>	
Cell Power and State	CALL[:CELL[1]]:POWer:SAMPlitude<numeric value>[<suffix>]	2
Cell BCH Number	CALL[:CELL[1]]:BCHannel[:ARFCn][:SELEcted]<numeric value> OR CALL[:CELL[1]]:BCHannel[:ARFCn]:<PGSM   EGSM   DCS   PCS> <numeric value>	3
Mobile Country Code	CALL[:CELL[1]]:MCCode <numeric value>	4
PCS Mobile Country Code	CALL[:CELL[1]]:PMNCode:VALue <numeric value>	4
Use PCS MNC	CALL[:CELL[1]]:PMNCode:STATe <ON   1   OFF   0>	4
PCS Mobile Country Code and Use PCS NMC State	CALL[:CELL[1]]:PMNCode[:SVALue] <numeric value>	4, 5
Mobile Network Code	CALL[:CELL[1]]:MNCCode <numeric value>	4
Location Area Code	CALL[:CELL[1]]:LACode <numeric value>	4
Network Color Code	CALL[:CELL[1]]:NCCode <numeric value>	4
Base Station Color Code	CALL[:CELL[1]]:BCCode <numeric value>	4
Paging IMSI	CALL:PAGing:IMSI <string>	
Repeat Paging State	CALL:PAGing:REPeat[:STATe] <ON   1   OFF   0>	
Paging Mode	CALL:PAGing:MODE <NORMAl   REORg>	7
Paging Multiframes	CALL:PAGing:MFRames <numeric value>	
Auto IMEI Request	CALL:IMEI:AUTO <ON   1   OFF   0>	
BA Table Entries	CALL[:CELL[1]]:BA:TABLE[:SELEcted][<numeric value>{,<numeric value>}] OR CALL[:CELL[1]]:BA:TABLE:<PGSM   EGSM   DCS   PCS> [<numeric value>{,<numeric value>}]	6

**Table Footnotes**

- 1 The broadcast band setting becomes the selected (:SElected) band (see note 3).
- 2 Sets amplitude to <numeric value> and state to ON in one command.
- 3 Sets the BCH channel for the broadcast band selected with the broadcast band command (see note 1).
- 4 Can only be set when Cell Activated State = OFF. See ["Things That Can Go Wrong" on page 23](#).
- 5 Sets PCS mobile country code to <numeric value> and state to ON in one command.
- 6 Sets the BA table entries for the broadcast band selected with the broadcast band command (see note 1).
- 7 Setting Paging Mode to Normal causes the MS to use discontinuous reception (that is, DRX = ON).

**Example 1. Programming Example:**

The following program example illustrates proper use of the BSE BCH configuration commands. Not all parameters are accessed. Note the use of the cell activated state command to set the network configuration parameters.

```
!*****
! Step 2: Configure Base Station Emulator (BSE)
!*****
!
OUTPUT Test_set;"CALL:CELL:BAND PGSM"
OUTPUT Test_set;"CALL:PAG:MODE REOR" ! Sets discontinuous reception to OFF
OUTPUT Test_set;"CALL:ACT OFF"
OUTPUT Test_set;"CALL:CELL:MCC 1;LAC 1;MNC 1;NCC 1;BCC 5"
OUTPUT Test_set;"CALL:ACT ON"
OUTPUT Test_set;"CALL:BCH 20"
OUTPUT Test_set;"CALL:POW:SAMP -85"
```

## Configuring the Traffic Channel Parameters

The traffic channel parameters are configured using the CALL processing subsystem commands shown in the following table.

**Table 2. Traffic Channel Settable Parameters**

Parameter	Command Syntax	Footnote
TCH Band	CALL:TCHannel:BAND <PGSM   EGSM   DCS   PCS>	1
Channel Number	CALL:TCHannel[:ARFCn][:SElected] <numeric value> OR CALL:TCHannel[:ARFCn]:<PGSM   EGSM   DCS   PCS> <numeric value>	2
Loopback Mode	CALL:TCHannel:LOOPback <OFF   A   B   C>	
Timeslot	CALL:TCHannel:TSLot <numeric value>	
Downlink Speech Source	CALL:TCHannel:DOWNlink:SPEech <NONE   ECHO   PRBS15   SIN300   SIN1000   SIN3000>	

### Table Footnotes

- 1 The TCH band setting becomes the selected band (see Note 2).
- 2 Sets the TCH channel for the TCH band selected with the TCH Band command (see Note 1).

### Example 2. Programming Example:

The following program example illustrates proper use of the BSE TCH configuration commands. Not all parameters are accessed.

```
OUTPUT Test_set;"CALL:TCH 45"
OUTPUT Test_set;"CALL:TCH:TSL 4"
```

## Things That Can Go Wrong

### Trying to Set the MCC, MNC, LAC, NCC, or BCC While the Cell Activated State = ON

Trying to set any of the network configuration parameters while the cell is in the active state will generate the following error:

```
GSM operation rejected; Attempting to set <MCC|MNC|LAC|NCC|BCC> while generating a BCH
```

**Background** The network configuration parameters are encoded into the messaging broadcast on the BCH. Changing the network parameter values while the BCH is active would require the BCH to be stopped, and have the new values encoded, and then the BCH would have to be re-started. This would cause calls to be dropped or disrupt a MS camped to the cell. Consequently the network configuration parameters cannot be changed while the cell is active.

**Control of the Cell Activated State** The active/inactive state of the cell is controlled using the cell activated state command. This command is only used when the operating mode is set to active cell mode.

#### Example 3. Command Syntax:

```
CALL[:CELL[1]]:ACTivated[:STATe]<ON|1|OFF|0>
```

#### Example 4. Programming Example:

```
OUTPUT Test_set; "CALL:ACT ON"
```

## Effects of Activating and Deactivating the Cell

**Effects of Deactivating the Cell** Among others (refer to the test set's reference information for a complete listing of actions), setting the cell activated state to OFF causes the following actions to take place:

- The control program is no longer prevented from setting the following parameters: MCC, MNC, PCS MNC, Use PCS MNC, BCC, NCC and LAC.
- All signaling operations, uplink demodulation and downlink (BCH & TCH) generation are stopped.
- Any measurements that rely on uplink demodulation are aborted. No special error messages are generated.

**Effects of Activating the Cell** Among others (refer to the test set's reference information for a complete listing of actions), setting the cell activated state to ON causes the following actions to take place:

- The control program is prevented from setting the following parameters: MCC, MNC, PCS MNC, Use PCS MNC, BCC, NCC and LAC.
- If the cell activated state was previously OFF, the TDMA frame number of the BS emulator starts from zero, and a BCH is generated.
- If a TCH was present prior to setting cell activated state to OFF, the TCH is not reinstated.

## Step 3: Configure the Measurement Execution Parameters

### Background

Measurement execution parameters control the conditions under which a measurement operates. The general set of measurement execution parameters and their generic categories are as follows:

- Measurement Averaging (used by most measurements)
  - Multi-Measurement Count State
  - Multi-Measurement Count State
- Measurement Triggering (used by most measurements)
  - Trigger Arm
  - Trigger Source
  - Trigger Delay
  - Trigger Qualifier
- Measurement Synchronization (used by some measurements)
  - Burst Synchronization
- Measurement Timeouts (used by all measurements)
  - Measurement Timeout
  - Measurement Timeout State
- Measurement Specific (execution parameters specific to an individual measurement)

---

**NOTE** Not all measurements use all the execution parameters shown above. Additionally, some measurements have parameters that are specific to the measurement such as offset frequency lists or filter settings. Each measurement has its own set of parameters which are unique to it and have no affect on the execution of other measurements. Refer to the GPIB syntax listing for a detailed list of execution parameters for individual measurements.

---



## Overview

The SETup subsystem is used to configure measurement parameters. Each individual measurement parameter can be set and queried using the associated SETup subsystem command. The general hierarchy of the SETup subsystem command structure is as follows:

```
SETup:<meas-mnemonic>:<measurement parameter><parameter setting/value>
```

The following table shows the measurements available in the Agilent E1960A GSM mobile test application and their associated <meas-mnemonic> used in the SETup command syntax.

**Table 1. Measurement Mnemonics Used In The SETup Subsystem**

Measurement	<meas-mnemonic>
Transmit Power	TXPower
Power vs Time	PVTime
Phase & Frequency Error	PFERror
Output RF Spectrum	ORFSpectrum
Bit Error	BERRor
Fast Bit Error	FBERRor
Decoded Audio	DAUDio
Analog Audio	AAUDio
I/Q Tuning	IQTuning
Dynamic Power	DPOWer

## Configuring Measurement Averaging Parameters

### Multi-Measurement Count State Parameter

The Multi-Measurement Count State parameter is used to turn measurement averaging on and off.

#### Example 1. Command Syntax:

```
SETup:<meas-mnemonic>:COUNT:STATE <ON|1|OFF|0>
```

#### Example 2. Programming Example:

```
OUTPUT Test_set;"SET:PVT:COUN:STATE ON"
```

would turn measurement averaging ON for the power versus time measurement.

### Multi-Measurement Count Number Parameter

The Multi-Measurement Count Number parameter sets the number of measurement samples taken during each measurement cycle when the COUNT:STATE parameter is set to ON.

#### Example 3. Command Syntax:

```
SETup:<meas-mnemonic>:COUNT:NUMBER <numeric value>
```

#### Example 4. Programming Example:

```
OUTPUT Test_set;"SET:TXP:COUN:NUMB 10"
```

would set the number of averages to 10 for the transmit power measurement.

### Configuring Multi-Measurement Count State and Count Number Simultaneously

The multi-measurement count state can be set to ON and the multi-measurement count number can be set to some value using a single complex command.

#### Example 5. Command Syntax:

```
SETup:<meas-mnemonic>:COUNT[:SNUMBER] <numeric value>
```

#### Example 6. Programming Example:

```
OUTPUT Test_set;"SET:TXP:COUN:SNUM 10"
```

would set the multi-measurement count state to ON and set the number of averages to 10 for the transmit power measurement. Note that in this example the optional command mnemonic :SNUMBER has been included for purposes of clarity.

## Configuring Measurement Triggering Parameters

### Trigger Source Parameter

The Trigger Source parameter selects the source of the measurement trigger signal.

#### Example 7. Command Syntax:

```
SETup:<meas-mnemonic>:TRIGger:SOURce <AUTO|IMMediate|PROToCol|RISE>
```

#### Example 8. Programming Example:

```
OUTPUT Test_set;"SET:TXP:TRIG:SOUR AUTO"
```

would set the trigger source to AUTO for the transmit power measurement.

### Trigger Delay Parameter

The Trigger Delay parameter controls the delay between the trigger event (the point in time at which the trigger signal is received) and the start of sampling. Negative values indicate that the sampling should occur prior to the trigger event.

#### Example 9. Command Syntax:

```
SETup:<meas-mnemonic>:TRIGger:DElay <numeric value>[<suffix>]
```

#### Example 10. Programming Example:

```
OUTPUT Test_set;"SET:TXP:TRIG:DEL 10 US"
```

would set the trigger delay to 10  $\mu$ s for the transmit power measurement.

### Trigger Qualifier Parameter

The Trigger Qualifier parameter enables or disables automatic trigger re-arming following a trigger event which occurred when no valid signal (burst) was present.

#### Example 11. Command Syntax:

```
SETup:<meas-mnemonic>:TRIGger:QUALifier <ON|1|OFF|0>
```

#### Example 12. Programming Example:

```
OUTPUT Test_set;"SET:TXP:TRIG:QUAL ON"
```

would turn the trigger qualifier on for the transmit power measurement.

### Step 3: Configure the Measurement Execution Parameters

#### Trigger Arm Parameter

The Trigger Arm parameter determines whether a measurement will make one measurement then stop (single), or automatically re-arm upon completion of one measurement and repeat the process (continuous).

#### Example 13. Command Syntax:

```
SETup:<meas-mnemonic>:CONTinuous <ON|1|OFF|0>
```

---

**NOTE** The recommend trigger arm setting for all measurements when using the remote user interface is single (CONTinuous OFF).

---

#### Example 14. Programming Example:

```
OUTPUT Test_set;"SET:TXP:CONT OFF"
```

would set the trigger arming to single for the transmit power measurement.

### Configuring the Burst Synchronization Parameter

#### Burst Synchronization Parameter

The burst synchronization parameter specifies where in the sampled data stream the measurement algorithm starts making its analysis of the captured data. Burst synchronization occurs after the measurement data is captured. The burst synchronization parameter's setting determines how the measurement's time reference is developed from the sampled data.

Not all measurements will have synchronization choices and not all synchronization choices will be available in measurements that use synchronization. Measurement synchronization and measurement triggering are independent settings and may be used in any combination.

#### Example 15. Command Syntax:

```
SETup:<meas-mnemonic>:BSYNc <MIDamble|AMPLitude|NONE>
```

#### Example 16. Programming Example:

```
OUTPUT Test_set;"SET:PVT:BSYN MID"
```

would set the burst synchronization to midamble for the power versus time measurement.

## Configuring Measurement Timeout Parameters

### Measurement Timeout State Parameter

The Measurement Timeout State parameter is used to enable or disable measurement timeout functionality.

#### Example 17. Command Syntax:

```
SETup:<meas-mnemonic>:TIMEout:STATE <ON|1|OFF|0>
```

#### Example 18. Programming Example:

```
OUTPUT Test_set;"SET:PVT:TIM:STAT ON"
```

would enable measurement timeouts for the power versus time measurement.

### Measurement Timeout Time Parameter

The Measurement Timeout Time parameter sets the maximum time that a measurement will execute before failing with a timeout error (when the TIMEout:STATe parameter is set to ON).

#### Example 19. Command Syntax:

```
SETup:<meas-mnemonic>:TIMEout:TIME <numeric value>[<suffix>]
```

#### Example 20. Programming Example:

```
OUTPUT Test_set;"SET:TXP:TIM:TIME 10 S"
```

would set the measurement timeout time to 10 seconds for the transmit power measurement.

### Configuring Measurement Timeout State and Timeout Time Simultaneously

The measurement timeout state can be set to ON and the measurement timeout time can be set to some value using a single complex command.

#### Example 21. Command Syntax:

```
SETup:<meas-mnemonic>:TIMEout[:STIME] <numeric value>[<suffix>]
```

#### Example 22. Programming Example:

```
OUTPUT Test_set;"SET:TXP:TIM:STIM 10"
```

would set the measurement timeout state to ON and set the measurement timeout time to 10 seconds for the transmit power measurement. Note that in this example the optional command mnemonic :STIME has been included for purposes of clarity.

## Configuring Measurement Specific Parameters

### Background

Some measurements have parameters that are specific to the measurement. Refer to the GPIB syntax listings for a detailed list of execution parameters for individual measurements. This section gives you some insight into the possible programming techniques that can be used to configure these measurement specific execution parameters.

### Sending Comma-Separated Parameter Configuration Lists to the Test Set

High-level measurements in the test application may require numerous parameters to configure the measurement. For example: the output RF spectrum measurement can require up to 22 frequency offsets for the modulation part of the measurement and up to 8 frequency offsets for the switching part of the measurement. The offsets are sent as comma separated lists. There are a variety of techniques that can be used to send these lists. Some of these techniques are shown below.

1. Include each individual parameter in the command itself. For example:

```
OUTPUT Test_set;"SET:ORFS:SWIT:FREQ .4MHZ,.6MHZ,-.4MHZ,-.6MHZ"
```

2. Store the parameter values in a data structure and send the command with the data structure appended to it. For example:

- Using a string variable:

```
DIM Swit_offs$(255)
Swit_offs$=".4MHZ,.6MHZ,-.4MHZ,-.6MHZ,1.2MHZ,-1.2MHZ"
OUTPUT Test_set;"SET:ORFS:SWIT:FREQ "&Swit_offs
```

- Using numeric arrays:

```
OPTION BASE 1
REAL Swit_offs(8),Mod_offs(22)
!
DATA 400,-400,600,-600,1200,-1200,1800,-1800
DATA .1,-.1,.2,-.2,.25,-.25,.4,-.4,.6,-.6,.8,-.8
DATA 1,-1,1.2,-1.2,1.4,-1.4,1.6,-1.6,1.8,-1.8
!
READ Swit_offs(*)
READ Mod_offs(*)
!
Swit_img:IMAGE K,7(K,"KHZ,"),K,"KHZ"
Mod_img:IMAGE K,21(K,"MHZ,"),K,"MHZ"
OUTPUT Test_set USING Swit_img;"SET:ORFS:SWIT:FREQ",Swit_offs(*)
OUTPUT Test_set USING Mod_img;"SET:ORFS:MOD:FREQ",Mod_offs(*)
```

**Example 23. Programming Example:**

The following example illustrates configuring the measurement execution parameters for the output RF spectrum, transmit power, and phase and frequency error measurements.

```

!*****
! Step 3: Configure Measurement Execution Parameters
!*****
!
! Configure ORFS Measurement:
!
OUTPUT Test_set;"SET:ORFS:SWIT:COUN 5"      ! Examples of using complex
OUTPUT Test_set;"SET:ORFS:MOD:COUN 10"     ! commands to set multi-meas
                                           ! state and count at same time.
OUTPUT Test_set;"SET:ORFS:TRIG:SOUR AUTO"  ! Set trig source to AUTO.
OUTPUT Test_set;"SET:ORFS:CONT OFF"       ! Set trig mode to single.
OUTPUT Test_set;"SET:ORFS:TIM 60"        ! Set timeout time to 60 sec.
! Put switching and modulation offsets to be tested into string variables.
Swit_offs$="400KHZ,-400KHZ,600KHZ,-600KHZ,1200KHZ,-1200KHZ,1800KHZ,-1800KHZ"
Mod_offs$=".2MHZ,-.2MHZ,.4MHZ,-.4MHZ,.6MHZ,-.6MHZ,.8MHZ,-.8MHZ,1MHZ,-1MHZ"
OUTPUT Test_set;"SET:ORFS:SWIT:FREQ "&Swit_offs$
OUTPUT Test_set;"SET:ORFS:MOD:FREQ "&Mod_offs$
!
! Configure TX Power Measurement:
!
OUTPUT Test_set;"SET:TXP:COUN 3"
OUTPUT Test_set;"SET:TXP:TRIG:SOUR RISE;QUAL ON"
OUTPUT Test_set;"SET:TXP:CONT OFF"
OUTPUT Test_set;"SET:TXP:TIM 20"
!
! Configure Phase & Frequency Error Measurement:
!
OUTPUT Test_set;"SET:PFER:COUN 8"
OUTPUT Test_set;"SET:PFER:TRIG:SOUR PROT;QUAL ON"
OUTPUT Test_set;"SET:PFER:CONT OFF"
OUTPUT Test_set;"SET:PFER:TIM 30"
OUTPUT Test_set;"SET:PFER:BSYN MID

```

## Step 4: Establish an Active Link with Mobile Station

### Background

#### Call Connect/Disconnect Synchronization

When the control program requires that an active link be established/terminated between the mobile station and the test set, the commands necessary to initiate the call connect/disconnect process are sent to the test set (for a BS originated/terminated call) or to the mobile station (for a MS originated/terminated call). In either case, synchronization is defined as the control program being able to empirically determine when the call has been successfully connected/disconnected so that the control program can proceed, or being able to empirically determine that the call has not been successfully connected/disconnected so that the control program can take appropriate action.

The determination is made by monitoring the call state as the call connect/disconnect process progresses.

#### Call States

At any instant in time a call can be in one of the following states:

- Idle
- Setup Request
- Proceeding
- Alerting
- Disconnecting
- Connected

Setup Request, Proceeding, Alerting and Disconnecting are referred to as transitory states because the amount of time which the call can spend in any of these states is limited by GSM protocol (that is, the call transitions through these states, it is not allowed to stay in a transitory state forever).

---

**NOTE**      If repeat paging is on it is possible for the call process to stay in one of the transitory states beyond the time specified by the GSM protocol timers.

---

The control program can directly query the state of a call with the CALL:STATus:STATE? query command, which immediately returns the current call state (that is, Idle, Setup Request, Proceeding, Alerting, Disconnecting, or Connected)



## Determining if a Call Connect/Disconnect Process is Completed

The most common technique used by control programs to determine if a call connect/disconnect process has completed (either successfully or unsuccessfully), is to repeatedly query the call state using the CALL:STATus:STATE? query command inside a program loop. The return value from each query is checked to determine if the connect/disconnect process is proceeding or has reached the desired state.

There are, however, some inherent problems associated with this technique:

- The rapid polling of the instrument increases bus traffic and places increased demand on the instrument's processors to respond to the constant stream of queries.
- The control program must handle failure conditions. For example: if a call origination process is started but the call never leaves the Idle state, the control program must incorporate some technique to prevent the program from staying in the loop forever waiting for a transition out of the Idle state.

The test set implements a set of commands designed specifically for call connect/disconnect synchronization. (see [“Step 8: Disconnect the Mobile Station from the BSE” on page 57](#) for call disconnect synchronization). These commands directly address many of the inherent problems discussed above. When properly used these commands eliminate the need for rapid polling of the instrument, and relieve the programmer of many of the tasks associated with error handling.

## Call Connect/Disconnect Synchronization Commands

**Call Connected State Query Command** The call-connected-state query command is used to query the connected state of a call. This command allows the control program to determine if a call is connected (that is, in the Connected state) or disconnected (that is, in the Idle state), with a built-in provision to automatically wait if the call is in one of the transitory states.

The basic operation of this query is:

- If the call is in the Connected state when the query is received by the test set, the query immediately returns a 1.
- If the call is in the Idle state (that is, disconnected) when the query is received by the test set, the query immediately returns a 0.
- If the call is in one of the transitory states (that is, Setup Request, Proceeding, Alerting, or Disconnecting) when the query is received by the test set, the query hangs (that is, does not return an answer) until the call state changes to either Idle or Connected and then behaves as above.

The call-connected-state query command can be used at any time to determine the connected state of a call. The built-in provision to automatically wait if the call is in one of the transitory states eliminates the need for rapid polling when the call-connected-state query command is used to synchronize to a call connect/disconnect process.

---

**NOTE** If repeat paging is on, a call origination process can stay in one of the transitory states until the mobile either answers the page or until the user stops the paging process. This means that if a call-connected-state query command is sent to the test set with repeat paging set to on, the query could hang “forever”.

---

## Step 4: Establish an Active Link with Mobile Station

### Example 1. Command Syntax:

```
CALL:CONNECTed[:STATE]?
```

**Using the Call Connected State Query for Call Connect Synchronization** The call-connected-state query only hangs if the call is in a transitory state, otherwise it immediately returns a 1 (Connected state) or a 0 (Idle state). At the start of a call connect process the call state is Idle. Sending call-connected-state query at the start of a call connect process could immediately return a zero if the query is satisfied before the connection process has started (that is, moved from the Idle state into one of the transitory states). For correct call connect synchronization it is necessary that the query be temporarily held off until after the call connect process has started. A call-state-change-detector is provided which can be used to temporarily hold off the query from returning an answer until the appropriate state change has occurred.

**Call Connected Arm Command** The call-connected-arm command is used to 'arm' the call-state-change-detector.

### Example 2. Command Syntax:

```
CALL:CONNECTed:ARM[:IMMediate]
```

If the call-state-change-detector is armed when a call-connected-state query is received, the reply is held off until the call-state-change-detector is disarmed. The call-state-change-detector is disarmed upon a state change from any of the transitory states to the Idle or Connected state.

The call-state-change-detector is not disarmed by a state change from Idle to any of the transitory states, from Connected to any of the transitory states, nor is it disarmed by any transitions from Idle to Idle, or Connected to Connected. These restrictions ensure that when the call-connected-state query returns an answer:

- the connect process has started since the call state must have moved from Idle to one of the transitory states
- AND
- the connect process has finished since the call state has moved from a transitory state to either the Idle or Connected state.

The arm state of the change detector can be queried with the call-connected-arm-state query command. This query never hangs and immediately returns a 1 if the change detector is armed and a 0 if it is not armed. The command is:

### Example 3. Command Syntax:

```
CALL:CONNECTed:ARM:STATE?
```

**Using the Call Connected Arm Command for Call Connect Synchronization** The call-state-change-detector arm command is used by the control program to tell the test set that it is expecting a change to the state of a call prior to initiating the state change. By first arming the call-state-change-detector, then querying the call connected state, and then attempting a BS or MS originated call, the call-connected-state query will hang until the connection operation begins and then reaches a final (Idle or Connected) state.

However, if the change detector is armed and a call connection is attempted but the call state never progresses from the Idle state, the call-connected-state query would hang forever. This could easily happen if the mobile is badly broken, the mobile is not connected to the test set, no one pushes the “send” button on the mobile, etc.

A call-state-change-detector time-out timer is provided which is used to prevent the call-connected-state query from hanging forever.

**Call Connected Time-out Command** The call-connected-time-out command is used to set the time-out value for the call-state-change-detector time-out timer.

#### Example 4. Command Syntax:

```
CALL:CONNECTed:TIMEout <numeric value>[<suffix>]
```

**Using the Call State Change Detector Time-out for Call Connect Synchronization** The call-state-change-detector time-out mechanism allows the test set to disarm the call-state-change-detector which releases the call connected state query if it is currently hanging.

The time-out timer is started whenever the call-state-change-detector is armed or gets rearmed when already armed. The duration of the time-out is set using the call-connected-time-out command and should be set to the maximum amount of time the control program should wait between arming and the connect process to begin. Once the process starts and the call state has moved into one of the transitory states the GSM defined protocol timers take over and prevent the call state from staying in a transitory state forever.

If the timer expires while the call is in the Idle or Connected state, the call-state-change-detector is disarmed, which releases the call connected state query if it is currently hanging.

If the timer expires while the call is in one of the transitory states it is ignored as, once in any transitory state, the GSM-defined protocol timers limit the amount of time that can be spent in any transitory state.

**Call-state-change-detector Auto Arming** As a programming convenience the test set automatically arms the call-state-change-detector, using a fixed time-out value of 60 seconds, whenever a BS originate or BS disconnect is requested.

Because of this, there is never a need for the control program to explicitly arm the call-state-change-detector or set a call-state-change-detector time-out value before BS initiated events. If for sake of coding efficiency, the programmer wishes to use the same code segment for both BS and MS call processing events, the commands to arm the call-state-change-detector and to set the call-state-change-detector time-out time will be accepted but ignored should the control program actually send the commands to the test set for BS call processing events.

## Overview

Establishing an active link with the mobile station when the test set is in active cell operating mode can be accomplished in one of two ways:

- Base station originated call
- Mobile station originated call

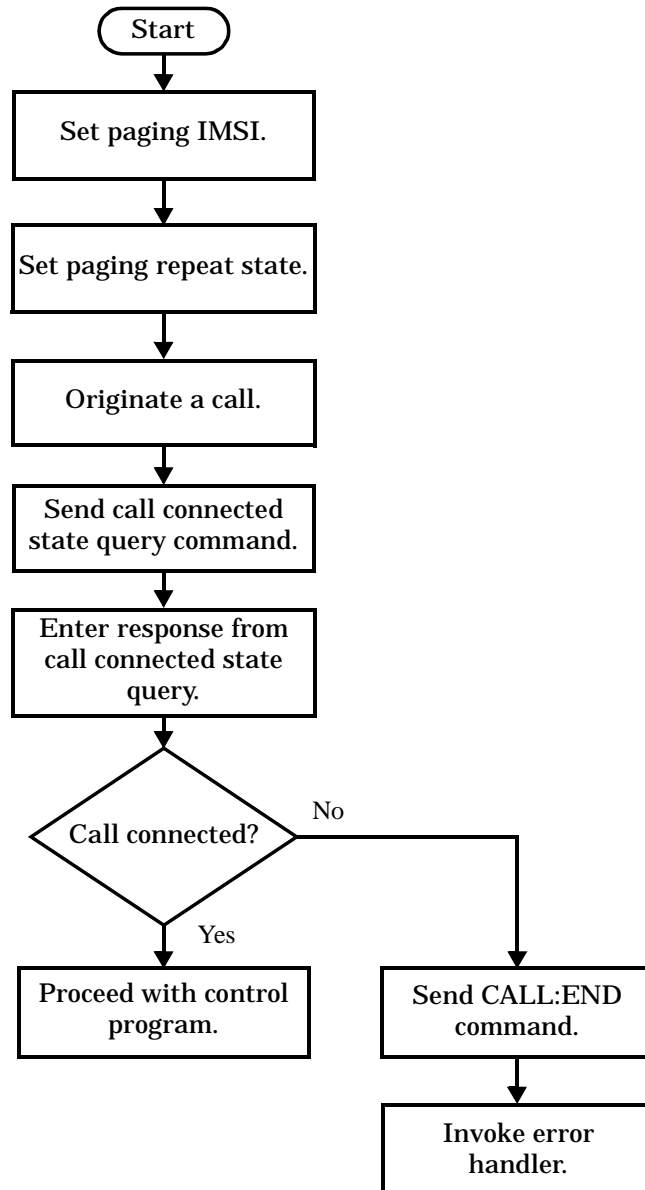
## Process for Making a Base Station Originated Call

The recommended process for making a base station originated call is shown in [“Step 4: Figure 1. Process for Making a Base Station Originated Call” on page 37.](#)

The CALL:ORIGINate command is used to initiate a base station originated call.

If the call origination process fails it is necessary to send the CALL:END command to the test set to force immediate termination of all processes associated with the current call origination. This ensures that if another CALL:ORIGINate command is sent to the test set before all processes associated with the failed call origination have been terminated, it will not be ignored. Note that if the test set is currently executing a call origination and it receives another call origination command it will be ignored (that is, you are telling the test set to do something it is already doing and hence it will accept the command but it will be ignored).

## Step 4: Figure 1. Process for Making a Base Station Originated Call



## Step 4: Establish an Active Link with Mobile Station

### Example 5. Programming Example:

```
!*****
! Step 4: Establish Active Link with Mobile Station
!*****
!
OUTPUT Test_set;"CALL:PAG:IMSI `001012345678901'" ! Set paging IMSI
OUTPUT Test_set;"CALL:PAG:REP OFF" ! Set paging repeat state to off
OUTPUT Test_set;"CALL:ORIG" ! Start a base station originated call
OUTPUT Test_set;"CALL:CONN:STAT?" ! Hanging GPIB query
ENTER Test_set;Call_connected ! Program will hang here until
! origination passes or fails
IF NOT Call_connected THEN ! Check if connection successful
    OUTPUT Test_set;"CALL:END"
! <put error handler here>
END IF
! Call is connected so proceed with control program
```

### Call Origination Process Commands

**Paging the Mobile Station** Paging the mobile station is accomplished using the CALL:ORIGinate command.

### Example 6. Command Syntax:

```
CALL:ORIGinate
```

### Example 7. Programming Example:

```
OUTPUT Test_set;"CALL:ORIG"
```

would start the process of making a base station originated call.

**Setting the Paging IMSI** The paging IMSI is set using the PAGIng:IMSI command.

### Example 8. Command Syntax:

```
CALL:PAGIng:IMSI <string>
```

**Example 9. Programming Example:**

```
OUTPUT Test_set;"CALL:PAG:IMSI `001012345678901`"
```

would set the paging IMSI to 001012345678901.

**Setting the Paging Repeat State** The paging repeat state is set using the PAGING:REPEAT:STATE command.

**Example 10. Command Syntax:**

```
CALL:PAGING:REPEAT[:STATE] <ON|1|OFF|0>
```

**Example 11. Programming Example:**

```
OUTPUT Test_set;"CALL:PAG:REP OFF"
```

would turn on paging repeat.

**Process for Making a Mobile Station Originated Call**

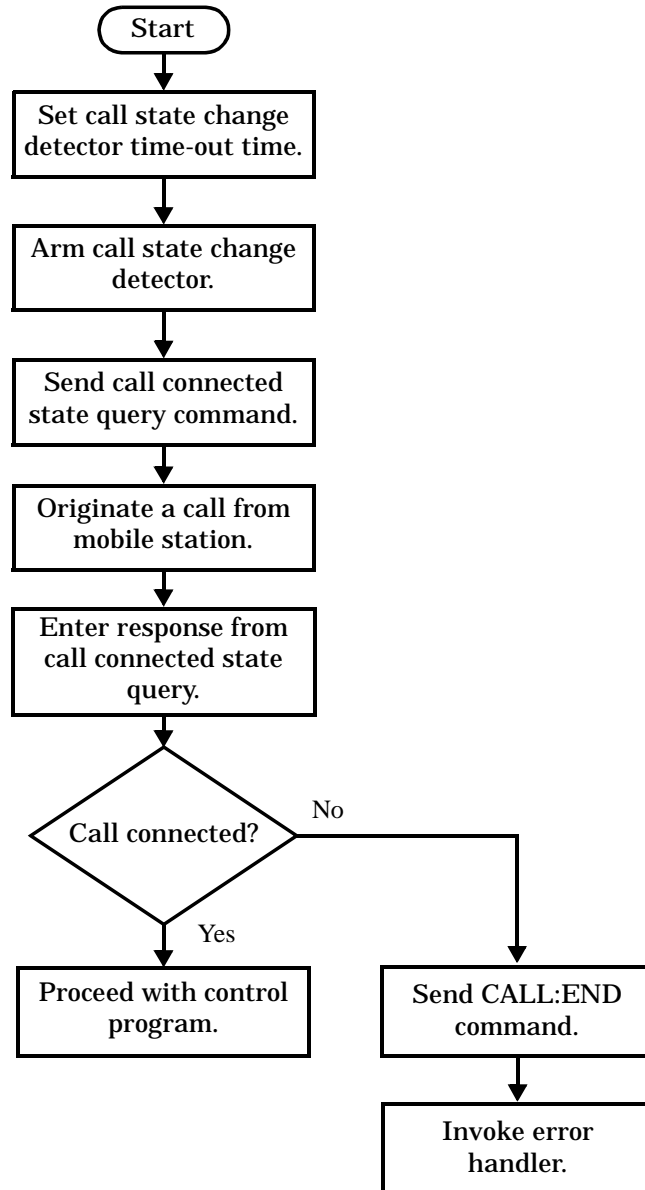
The recommended process for making a mobile station originated call is shown in [“Step 4: Figure 2. Process For Making A Mobile Station Originated Call”](#) on page 40.

There is no facility in the test set to initiate a call connect from the mobile station. This must be accomplished manually or through a test bus built into the mobile station.

If the call origination process fails it is necessary to send the CALL:END command to the test set to force immediate termination of all processes associated with the current call origination. This ensures that if the mobile station attempts another originate before all processes associated with the failed call origination have been terminated, it will not be ignored. Note that if the test set is currently executing a call origination and it receives another call origination command it will be ignored (that is, you are telling the test set to do something it is already doing and hence it will accept the command but it will be ignored).

For mobile station originated calls where the call is originated by physically dialing a number (as opposed to using a test bus) ensure that the call-state-change-detector time-out time is long enough to allow a human to dial the number.

Step 4: Figure 2. Process For Making A Mobile Station Originated Call





**Example 12. Programming Example:**

```
OUTPUT Test_set;"CALL:CONN:TIM 5"      ! Set timeout time to 5 seconds
OUTPUT Test_set;"CALL:CONN:ARM"        ! Arm the change detector
OUTPUT Test_set;"CALL:CONN:STAT?"      ! Initiate call connect state query
DISP "Originate call from mobile station."
ENTER Test_set;Call_connected          ! Program will hang here until
                                        ! origination passes or fails
IF NOT Call_connected THEN             ! Check if connection successful
    OUTPUT Test_set;"CALL:END"
! <put error handler here>
END IF
! Call is connected so proceed with control program
```

## Step 5: Set the Mobile Station's Operating Conditions

### Overview

The mobile station's operating conditions are set using the CALL processing subsystem commands shown in the following table.

**Table 1. Settable Mobile Station Operating Conditions**

Parameter	Command Syntax	Table Footnotes
Timing Advance	CALL:MS:TADVance <numeric value>	
Transmit Level	CALL:MS:TXLevel[:SElected] <numeric value> OR CALL:MS:TXLevel:<PGSM   EGSM   DCS   PCS> <numeric value>	1
Discontinuous Transmission	CALL:MS:DTX[:STATe] <ON   1   OFF   0>	

### Table Footnotes

1 The TCH band setting becomes the selected band.

### Example 1. Programming Example:

```
!*****
! Step 5: Set Mobile Station Operating Conditions
!*****
!
OUTPUT Test_set;"CALL:MS:DTX OFF"
OUTPUT Test_set;"CALL:MS:TXL 14
```

---

## Step 6: Make Measurements

### Background

The multiple signal path, DSP based, multiple processor architecture of the test set allows the test set to make concurrent measurements. This means that:

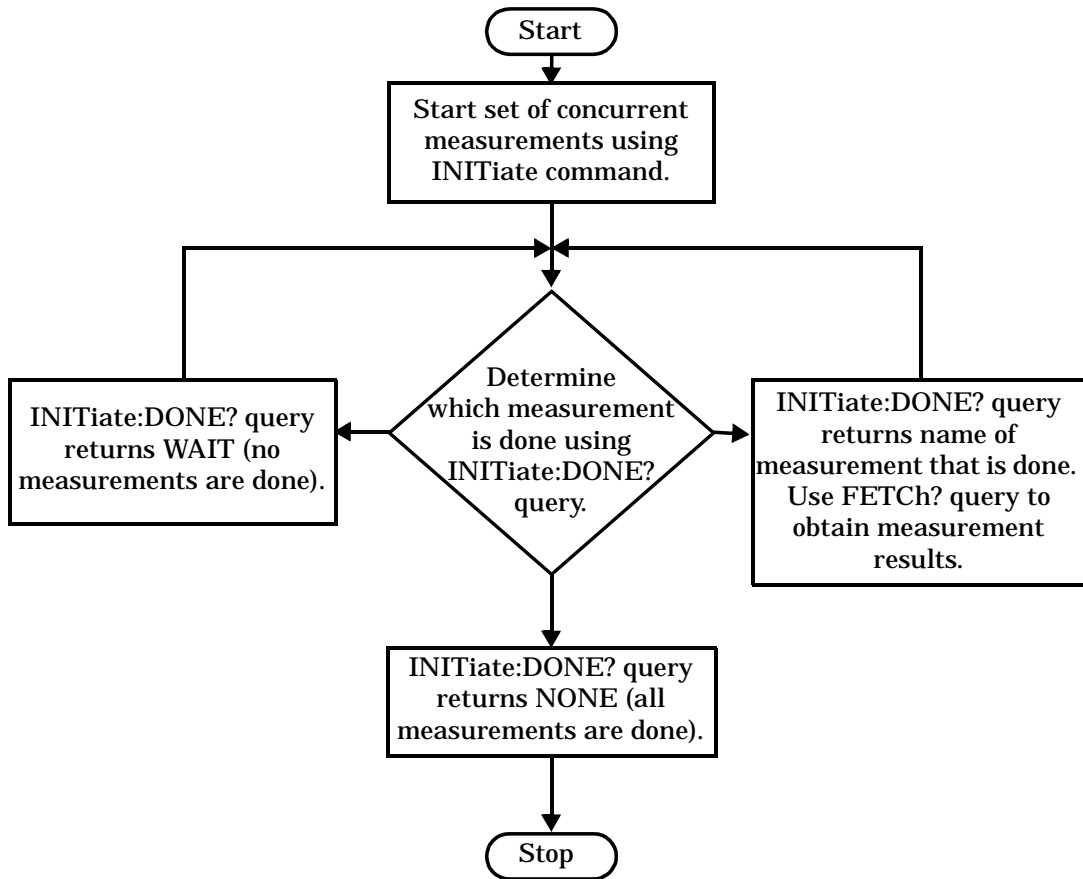
- multiple measurements can execute and finish at the same time (concurrently)
- individual measurement completion is not influenced by other measurement processes
- availability of measurement results is not dependent upon the sequence that the measurements were requested in
- results from measurements that take few processor cycles are available without having to wait for measurements that take many processor cycles

There are no special programming commands or techniques required to implement measurement concurrency.

“[Step 6: Figure 1. Process for Making Measurements](#)” on [page 44](#) shows the recommended process for making concurrent measurements using the test set’s command set.

Step 6: Make Measurements

Step 6: Figure 1. Process for Making Measurements



**Example 1. Programming Example:**

The following program segment illustrates making a transmit power measurement and a phase and frequency error measurement concurrently using the recommended process shown in “[Step 6: Figure 1. Process for Making Measurements](#)” on page 44.

```

!*****
! Step 6: Make Measurements
!*****
!
! Step 6a: Start Set of Concurrent Measurements:
!
OUTPUT Test_set;"INIT:TXP;PFER"
!
! Step 6b: Determine If A Measurement Is Done:
!
LOOP
  OUTPUT Test_set;"INIT:DONE?"
  ENTER Test_set;Meas_done$
!
! Step 6c: Obtain Measurement Results
!
  SELECT Meas_done$
    CASE "TXP"
      OUTPUT Test_set;"FETC:TXP:POW?"
      ENTER Test_set;Avg_tx_power
    CASE "PFER"
      OUTPUT Test_set;"FETC:PFER:RMS?"
      ENTER Test_set;Max_rms_phas_er
  END SELECT
EXIT IF Meas_done$ = "NONE"
END LOOP

```

## Things That Can Go Wrong

### Measurement Integrity Always Returns a Value of 6

**Background** A measurement integrity value of 6 indicates that some characteristic of the input signal is under range. Typically this will be the amplitude (power) of the DUT signal. This low amplitude will cause the level of the DSP sampler to be below a threshold required by the measurement algorithm to produce results of specified accuracy.

**Possible Cause** One of the most likely causes of a measurement underrange condition is DUT signal loss caused by fixture loss or cable loss.

**Suggested Workaround** Fixture loss or cable loss can be compensated for by using the RF IN/OUT port's amplitude offset parameter.

#### Example 2. Command Syntax:

```
SYSTem:CORRection:GAIN <numeric value>[<suffix>]
```

```
SYSTem:CORRection:STATe <1|ON|0|OFF>
```

Complex form of command (sets gain to <numeric value> and state to ON using single command):

```
SYSTem:CORRection:SGAin <numeric value>[<suffix>]
```

#### Example 3. Programming Example:

```
OUTPUT Test_set; "SYST:CORR:SGA -6"
```

would set the RF IN/OUT port's amplitude offset to -6 dB and set the correction state to ON.

## Step 6a: Start Set Of Concurrent Measurements

### Starting Measurements

The INITiate command is used to start measurements. Each individual measurement in a test application can be started using the INITiate command. For starting measurements, the syntax of the INITiate command is as follows:

#### Example 1. Command Syntax:

```
INITiate:<meas-mnemonic>[:ON]
```

The following table shows the measurements available in the Agilent Technologies E1960A GSM mobile test application and their associated <meas-mnemonic> used in the INITiate command syntax.

**Table 1. Measurement Mnemonics Used In The INITiate Subsystem**

Measurement	<meas-mnemonic>
Transmit Power	TXPower
Power vs Time	PVTime
Phase & Frequency Error	PFERror
Output RF Spectrum	ORFSpectrum
Bit Error	BERRor
Fast Bit Error	FBERRor
Decoded Audio	DAUDio
Analog Audio	AAUDio
I/Q Tuning	IQTuning
Dynamic Power	DPOWER

## Step 6a: Start Set Of Concurrent Measurements

### **Example 2. Programming Example:**

```
OUTPUT Test_set; "INIT:TXP"
```

would start the transmitter power measurement.

### **Using Compound Commands to Start Multiple Measurements**

More than one measurement can be started using a single INITiate command. For example:

```
OUTPUT Test_set; "INIT:TXP;PFER"
```

would start the transmit power measurement and the phase and frequency error measurement. These measurements would then run concurrently.



## Step 6b: Determine if a Measurement Is Done

### Background

After a set of concurrent measurements have been started, it is desirable that the control program be able to determine when individual measurement results are available so that the control program can request that measurement's results without having to wait on other measurements which have not yet completed.

### Overview

The INITiate:DONE? query command is used to determine which measurement is finished.

As the name implies, the query returns the name of whichever active measurement is done so that the control program can request that measurement's results.

This command is query only and returns only one response per query. The responses returned and their meaning are shown in the following table.

Once a measurement is reported as being done via the INITiate:DONE? query it is removed from the done list (measurements are only reported as being done once). The design of the INITiate:DONE? query is predicated on the control program immediately fetching a measurement's results once it is reported as being done.

**Table 1. Responses Returned from INITiate:DONE? Query**

Response	Meaning
TXP	The transmit power measurement is done.
PVT	The power versus time measurement is done.
PFER	The phase and frequency error measurement is done.
ORFS	The output RF spectrum measurement is done.
AAUD	The analog audio measurement is done.
DAUD	The decoded audio measurement is done.
BERR	The bit error measurement is done.
FBER	The fast bit error measurement is done.
DPOW	The dynamic power measurement is done.
IQT	The I/Q Tuning measurement is done.
WAIT	There are one or more measurements that are in progress, but none of those measurements are done yet.
NONE	No measurements are in progress.

Step 6b: Determine if a Measurement Is Done

**Example 1. Command Syntax:**

INITiate:DONE?

**Example 2. Programming Example:**

See [“Programming Example:”](#) on page 45.

---

## Step 6c: Obtain a Set of Measurement Results

### Background

In order to minimize bus traffic in the manufacturing environment the test set's high-level measurements have been designed to return multiple measured values in response to a single measurement request.

For example: if a transmit power measurement with averaging is initiated there will be five measurement results available as follows:

1. Measurement integrity value
2. Average value
3. Minimum value
4. Maximum value
5. Standard deviation value

The test set has been designed with the capability to return the measurement results in a variety of formats to suit the needs of the measurement environment. For example, the transmitter power measurement results can be returned as:

- Measurement integrity and average value  
OR
- Average value and minimum value and maximum value and standard deviation value  
OR
- Average value only  
OR
- Minimum value only  
OR
- Maximum value only  
OR
- Standard deviation value only  
OR
- Measurement integrity value only

The formats available for individual measurements can be found in the test set's FETCh? subsystem's GPIB command syntax reference information.

## Overview

The FETCh subsystem is used to query measurement results. The measurement results from each measurement in a test application can be queried using the FETCh subsystem. The general hierarchy of the FETCh command structure is as follows:

```
FETCh:<meas-mnemonic>:<result format>?
```

The following table shows the measurements available in the Agilent Technologies E1960A GSM mobile test application and their associated <meas-mnemonic> used in the FETCh command syntax.

The command syntax used to obtain the various measurement result formats (<result format>) for each measurement can be found in the test set's FETCh? subsystem's GPIB command syntax reference information.

**Table 1. Measurement Mnemonics Used In The FETCh Subsystem**

Measurement	<meas-mnemonic>
Transmit Power	TXPower
Power vs Time	PVTime
Phase & Frequency Error	PFERror
Output RF Spectrum	ORFSpectrum
Bit Error	BERRor
Fast Bit Error	FBERror
Decoded Audio	DAUDio
Analog Audio	AAUDio
I/Q Tuning	IQTuning
Dynamic Power	DPOWer

### Example 1. Command Syntax:

```
FETCh:<meas-mnemonic>:<result format>?
```

### Example 2. Programming Example:

```
OUTPUT Test_set; "FETCh:TXP:POW:MIN?"
```

would return the minimum value from the set of samples taken during the transmit power measurement (when averaging is turned on and number of samples taken >1).

---

## Step 7: Perform an Intra-Cell Handover

### Background

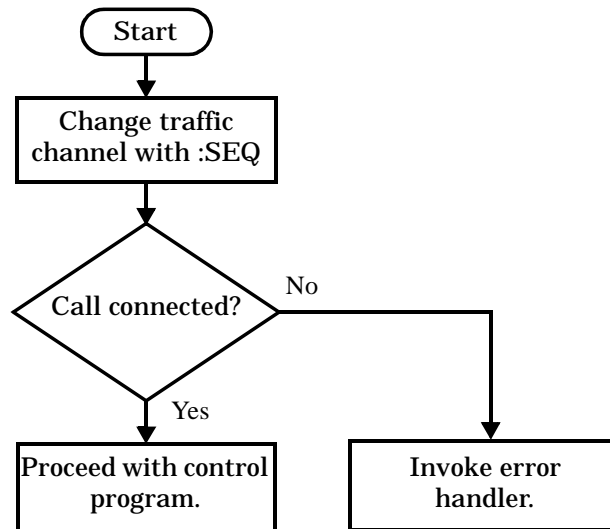
A handover is defined as assigning the mobile station to a new traffic channel. The test set is capable of performing two types of handovers:

- Intra-cell handover: assigning the mobile station to a new traffic channel within the currently active broadcast band.
- Dual-band handover: assigning the mobile station to a traffic channel in a traffic band which is different from the currently active traffic band.

### Performing an Intra-Cell Handover

An intra-cell handover is accomplished using the CALL:TCHannel command in conjunction with the :SEQ synchronization command. The recommended process for performing an intra-cell handover is shown in the following figure.

Step 7: Figure 1. Process for Performing an Intra-Cell Handover



## Step 7: Perform an Intra-Cell Handover

### Example 1. Command Syntax:

```
CALL:TCHannel[:ARFCn][:SElected]:SEQ <numeric value>
```

OR

```
CALL:TCHannel[:ARFCn]:<PGSM|EGSM|DCS|PCS>:SEQ <numeric value>
```

### Example 2. Programming Example:

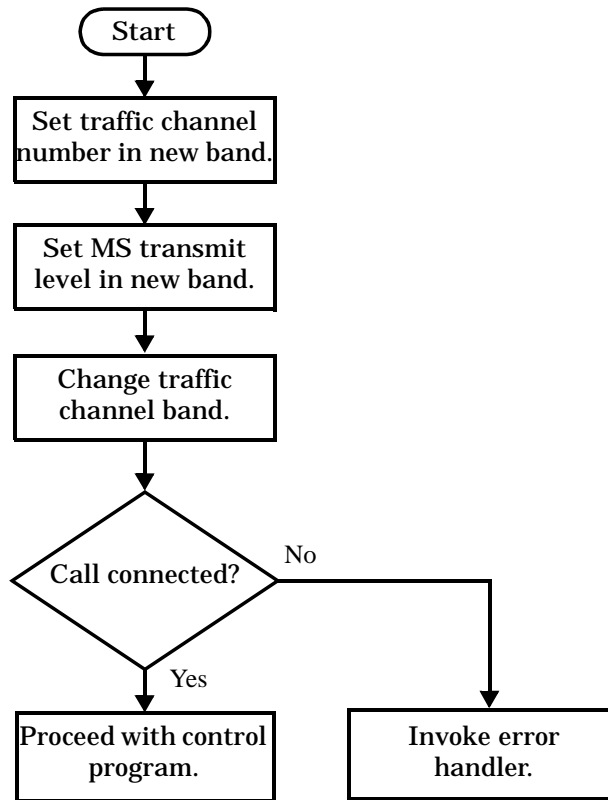
The following example illustrates how to use these commands to perform an intra-cell handover.

```
! existing conditions: a mobile station is connected to the test
! set, operating mode is set to active cell and a call is in the
! connected state.
! Step 1: Change the traffic channel number
OUTPUT Test_set;"CALL:TCH:SEQ 65"!Starts process of handing over MS
                                     !to new traffic channel 65.
                                     !No other commands will be processed
                                     !until this operation completes
                                     !because the :SEQ has been attached.
! Step #2: Check that the call is still in the connected state. It
           ! is possible that the MS did not successfully connect on the
           ! new channel.
OUTPUT Test_set;"CALL:STAT:STAT?"
ENTER Test_set;Call_status$
IF Call_status$ <> "CONN" THEN
! <put error handler here>
END IF
! Call is connected so proceed with control program
```

## Performing a Dual-Band Handover

A dual-band handover is accomplished using the CALL:TCHannel:BAND command. The recommended process for performing a dual band handover is shown in the following figure.

## Step 7: Figure 2. Process for Performing a Dual-Band Handover



## Step 7: Perform an Intra-Cell Handover

### Example 3. Programming Example:

The following example illustrates how to use the CALL:TCHannel:BAND command to perform a dual-band handover.

```
! existing conditions: a mobile station is connected to the test
! set, MS TX Level = 11, Timeslot = 4, Timing Advance = 0,
! operating mode is set to active cell, a call is in the
! connected state, and active broadcast band is EGSM
! Step #1: Configure the traffic channel in the new broadcast band
OUTPUT Test_set;"CALL:TCH:DCS 556"
OUTPUT Test_set;"CALL:MS:TXL:DCS 4"
! Step #2: Change the traffic channel band
OUTPUT Test_set;"CALL:TCH:BAND DCS" !This is a sequential command so no
                                     !other commands will be executed until
                                     !the handover is complete (the
                                     !MS has communicated to the BSE that it
                                     !has successfully transitioned to the
                                     !new channel OR a protocol timer has
                                     !timed out).
! Step #3: Check that the call is still in the connected state. It
           ! is possible that the MS did not successfully connect on the
           ! new channel.
OUTPUT Test_set;"CALL:STAT:STAT?"
ENTER Test_set;Call_state$
IF Call_state$ <> "CONN" THEN
! <put error handler here>
END IF
! Call is connected so proceed with control program
```



---

## Step 8: Disconnect the Mobile Station from the BSE

### Background

See “Step 4: Establish an Active Link with Mobile Station” for a discussion of call connect/disconnect synchronization.

### Using the Call Connected State Query for Call Disconnect Synchronization

The call-connected-state query only hangs if the call is in a transitory state, otherwise it immediately returns a 1 (Connected state) or a 0 (Idle state). At the start of a call disconnect process the call state is Connected. Sending a call-connected-state query at the start of a call disconnect process could immediately return a one if the query is satisfied before the disconnection process has started (that is, moved from the Connected state into one of the transitory states). For correct call disconnect synchronization it is necessary that the query be temporarily held off until after the call disconnect process has started. The call-state-change-detector is provided which can be used to temporarily hold off the query from returning an answer until the appropriate state change has occurred.

### Using the Call Connected Arm Command for Call Disconnect Synchronization

The call-state-change-detector arm command is used by the control program to tell the test set that it is expecting a change to the state of a call prior to initiating the state change. By first arming the call-state-change-detector, then querying the call connected state, and then attempting a BS or MS call termination, the call-connected-state query will hang until the disconnection operation begins and then reaches a final (Idle or Connected) state.

However, if the change detector is armed and a call disconnection is attempted but the call state never progresses from the Connected state, the call-connected-state query would hang forever. This could easily happen if the mobile is badly broken, no one pushes the “end” button on the mobile, etc.

The call-state-change-detector time-out timer is provided which is used to prevent the call-connected-state query from hanging forever.

### Using the Call State Change Detector Time-out for Call Disconnect Synchronization

The call-state-change-detector time-out mechanism allows the test set to disarm the call-state-change-detector which releases the call connected state query if it is currently hanging.

The time-out timer is started whenever the call-state-change-detector is armed or gets rearmed when already armed. The duration of the time-out is set using the call-connected-time-out command and should be set to the maximum amount of time the control program should wait between arming and the disconnect process to begin. Once the process starts and the call state has moved into one of the transitory states the GSM defined protocol timers take over and prevent the call state from staying in a transitory state forever.

If the timer expires while the call is in the Idle or Connected state, the call-state-change-detector is disarmed, which releases the call connected state query if it is currently hanging.

If the timer expires while the call is in one of the transitory states it is ignored as, once in any transitory state, the GSM-defined protocol timers limit the amount of time that can be spent in any transitory state.

## Overview

Terminating an active call with the mobile station when the test set is in active cell operating mode can be accomplished in one of two ways:

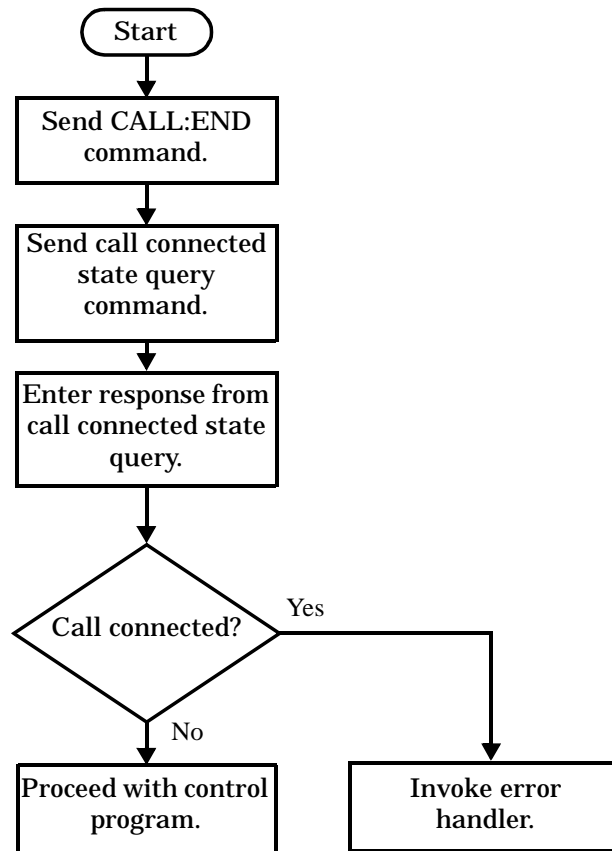
- Terminate the active call from the base station emulator
- Terminate the active call from the mobile station

## Terminating an Active Call from the Base Station Emulator

The recommended process for terminating an active call from the base station emulator is shown in the following figure.

The CALL:END command is used to initiate a base station disconnect.

Step 8: Figure 1. Process for Terminating an Active Call from the BSE



**Example 1. Programming Example:**

```

!*****
! Step 8: Disconnect Mobile Station From BSE
!*****
!
OUTPUT Test_set;"CALL:END"           ! Initiate a base station disconnect.
OUTPUT Test_set;"CALL:CONN:STAT?"   ! Initiate call connect state query.
ENTER Test_set;Call_connected       ! Program will hang here until state
                                     ! change or timer expires.
IF Call_connected THEN              ! Check if disconnect successful
! <put error handler here>
END IF
! Call is disconnected so proceed with control program

```

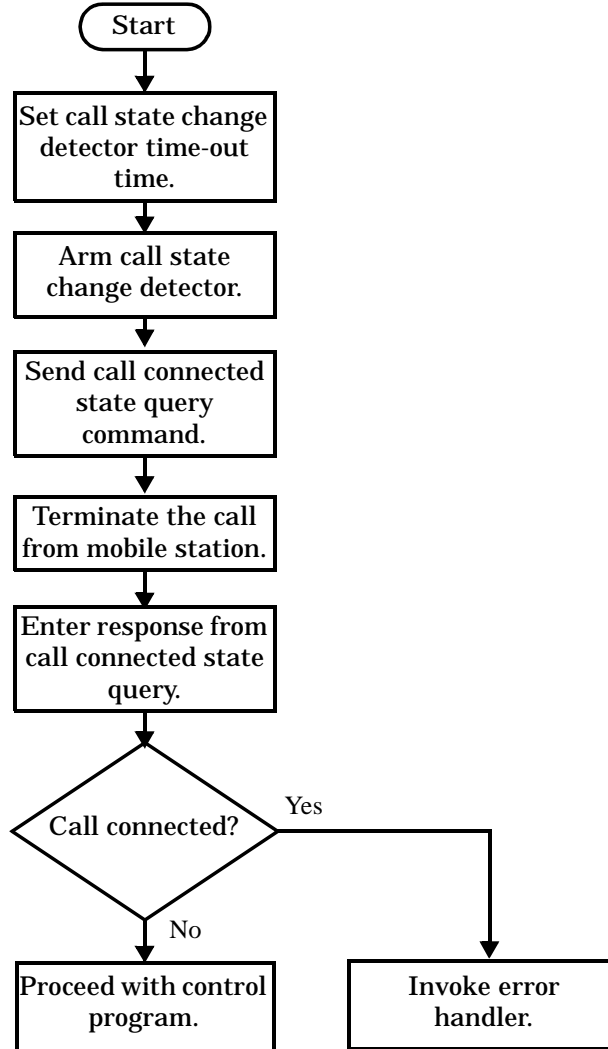
**Terminating an Active Call from the Mobile Station**

The process for terminating an active call from the mobile station is shown in the following figure.

There is no facility in the test set to initiate a call disconnect from the mobile station. This must be accomplished manually or through a test bus built into the mobile station.

For mobile station terminated calls where the call is terminated by physically pushing a button on the phone (as opposed to using a test bus) ensure that the call-state-change-detector time-out time is long enough to allow a human to push the button.

Step 8: Figure 2. Process for Terminating an Active Call from the Mobile Station



**Example 2. Programming Example:**

```
OUTPUT Test_set;"CALL:CONN:TIM 5" !Set timeout time to 5 seconds.
OUTPUT Test_set;"CALL:CONN:ARM"   !Arm the change detector.
OUTPUT Test_set;"CALL:CONN:STAT?" !Initiate call connect state query.
DISP "Terminate the call from the mobile station."
ENTER Test_set;Call_connected     !Program will hang here until state
                                   !change or timer expires.
IF Call_connected THEN             !Check if disconnect successful.
! <put error handler here>
END IF
! Call is disconnected so proceed with control program
```

## Step 8: Disconnect the Mobile Station from the BSE

---

## Comprehensive Program Example

This section presents two example programs for making measurements using the test set. The first program follows the task flow presented at the beginning of the programming note (see [“Figure 1. Typical Flow of Tasks Performed by Control Program” on page 15](#)) and which is discussed throughout the programming guide. The second program, [“Example Program Without Comments” on page 69](#), is basically the same as the first but comments have been removed and the coding reflects the use of compound commands and complex commands to achieve coding efficiency.

**Example Program With Comments**

```

10  ! Prog Name: com_man_ex.txt      Rev: A.0.2      Date Code: 12/18/98
20  !
30  ! Configure the BASIC environment, dimension and initialize variables.
40  ! These actions are unrelated to programming the Agilent 8960.
50  !
60  OPTION BASE 1
70  COM /Address/ Test_set
80  ! Allocate arrays to hold ORFS switching & modulation frequency offsets.
90  DIM Swit_offs$(255),Mod_offs$(255)
100 ! Allocate arrays to hold measurement results.
110 REAL Txpower(4)
120 Test_set=714 ! Test set's GPIB address.
130 PRINTER IS CRT
140 CLEAR SCREEN
150 !
160 ! Reset test set to start from a known state. Not always necessary to do full
170 ! preset in a manufacturing environment but desireable in programming example.
180 !
190 OUTPUT Test_set;"*RST"
200 !
210 ! Turn on the GPIB debugger. This is optional but very helpful for debugging
220 ! GPIB commands when developing new code.
230 !
240 OUTPUT Test_set;"SYST:COMM:GPIB:DEB:STAT ON"
250 !
260 ! Check error message queue and STOP if any errors present. This ensures that
270 ! the example program starts with no error conditions present in the test set.
280 !
290 CALL Chk_err_msg_que
300 !
310 !*****
320 ! Step 1: Set Test Set's Operating Mode to Active Cell
330 !*****
340 !
350 OUTPUT Test_set;"CALL:OPER:MODE CELL"
360 !
370 !*****
380 ! Step 2: Configure the Base Station Emulator (BSE)
390 !*****
400 !
410 ! Set RF IN/OUT port's amplitude offset to compensate for fixture loss of MS.
420 ! After setting offset, cell power settings reflect RF power at the MS antenna
430 ! input.
440 !
450 OUTPUT Test_set;"SYST:CORR:SGA -6" ! MS has a -6 dB fixture loss.
460 !
470 OUTPUT Test_set;"CALL:CELL:BAND PGSM" ! Set active broadcast band to PGSM.
480 OUTPUT Test_set;"CALL:ACT OFF" ! Deactivate cell to set network parms.
490 OUTPUT Test_set;"CALL:CELL:MCC 1;LAC 1;MNC 1;NCC 1;BCC 5" ! Set network parms.
500 OUTPUT Test_set;"CALL:ACT ON" ! Reactivate the cell.
510 OUTPUT Test_set;"CALL:BCH 20" ! Set broadcast channel to 20.

```



```

520 OUTPUT Test_set;"CALL:POW:SAMP -85" ! Set cell power to -85 dBm and cell
530 ! power state to ON with complex command.
540 OUTPUT Test_set;"CALL:TCH 45" ! Set traffic channel to 45.
550 OUTPUT Test_set;"CALL:TCH:TSL 4" ! Set timeslot to 4.
560 !
570 !*****
580 ! Step 3: Configure the Measurement Execution Parameters
590 !*****
600 !
610 ! Configure ORFS Measurement:
620 !
630 OUTPUT Test_set;"SET:ORFS:SWIT:COUN 5" ! Examples of using complex commands to
640 OUTPUT Test_set;"SET:ORFS:MOD:COUN 10" ! set multi-meas state and count at
650 ! same time.
660 OUTPUT Test_set;"SET:ORFS:TRIG:SOUR AUTO" ! Set trig source to AUTO.
670 OUTPUT Test_set;"SET:ORFS:CONT OFF" ! Set trig mode to single.
680 OUTPUT Test_set;"SET:ORFS:TIM 60" ! Set timeout time to 60 seconds.
690 ! Put switching and modulation offsets to be tested into string variables.
700 Swit_offs$="400KHZ,-400KHZ,600KHZ,-600KHZ,1200KHZ,-1200KHZ,1800KHZ,-1800KHZ"
710 Mod_offs$=".2MHZ,-.2MHZ,.4MHZ,-.4MHZ,.6MHZ,-.6MHZ,.8MHZ,-.8MHZ,1MHZ,-1MHZ"
720 OUTPUT Test_set;"SET:ORFS:SWIT:FREQ "&Swit_offs$
730 OUTPUT Test_set;"SET:ORFS:MOD:FREQ "&Mod_offs$
740 !
750 ! Configure TX Power Measurement:
760 !
770 OUTPUT Test_set;"SET:TXP:COUN 3"
780 OUTPUT Test_set;"SET:TXP:TRIG:SOUR RISE;QUAL ON"
790 OUTPUT Test_set;"SET:TXP:CONT OFF"
800 OUTPUT Test_set;"SET:TXP:TIM 20"
810 !
820 ! Configure Phase & Frequency Error Measurement:
830 !
840 OUTPUT Test_set;"SET:PFER:COUN 8"
850 OUTPUT Test_set;"SET:PFER:TRIG:SOUR PROT;QUAL ON"
860 OUTPUT Test_set;"SET:PFER:CONT OFF"
870 OUTPUT Test_set;"SET:PFER:TIM 30"
880 OUTPUT Test_set;"SET:PFER:BSYN MID"
890 !
900 !*****
910 ! Step 4: Establish an Active Link with the Mobile Station
920 !*****
930 !
940 OUTPUT Test_set;"CALL:PAG:IMSI '001012345678901'" ! Set paging IMSI.
950 OUTPUT Test_set;"CALL:PAG:REP OFF" ! Set paging repeat state to off.
960 !
970 ! This example uses a BSE originated call. The MS must be camped to the BSE
980 ! in order for the BSE to originate a call. The following code will try to
990 ! originate a call 50 times and then STOP the program. This should give
1000 ! adequate time for the MS to camp to the BSE.
1010 !
1020 ! NOTE: This technique will cause the following error to be displayed on the
1030 ! test set's display and be put in the error message queue each time
1040 ! that the call fails to connect. This is normal for this technique.

```

## Comprehensive Program Example

```
1050 ! 'GSM call disconnected; No response to page (Timer T3113 expiry)'  
1060 !  
1070 Tries=1  
1080 LOOP  
1090   OUTPUT Test_set;"CALL:ORIG"           ! Originate a call.  
1100   OUTPUT Test_set;"CALL:CONN:STAT?"     ! CALL:CONNected hanging GPIB query.  
1110   ENTER Test_set;Call_connected        ! Program will hang here until origination  
1120                                       ! process completes.  If successful and  
1130                                       ! the call is connected the query will  
1140                                       ! return a 1.  If unsuccessful and the call  
1150                                       ! is not connected the query returns 0.  
1160 EXIT IF Call_connected  
1170   OUTPUT Test_set;"CALL:END"  
1180   IF Tries=50 THEN  
1190     BEEP  
1200     DISP ""  
1210     PRINT "Call did not connect after";Tries;". Program terminated."  
1220     STOP  
1230   END IF  
1240   DISP "Call has not connected after";Tries;"attempts. Trying again."  
1250   Tries=Tries+1  
1260 END LOOP  
1270 DISP ""  
1280 !  
1290 !*****  
1300 ! Step 5: Set the Mobile Station's Operating Conditions  
1310 !*****  
1320 !  
1330 OUTPUT Test_set;"CALL:MS:DTX OFF"       ! Turn DTX off for all MS tests.  
1340 !  
1350 FOR Traf_chan=120 TO 124 STEP 2         ! Test channels 120, 122 & 124.  
1360   OUTPUT Test_set;"CALL:TCH:SEQ ";Traf_chan ! Use :SEQ to force sequential  
1370                                       ! execution of the TCH command.  
1380   OUTPUT Test_set;"CALL:STAT:STAT?"     ! Verify that the call is still in  
1390   ENTER Test_set;Call_status$          ! the connected state after handover.  
1400   IF Call_status$<>"CONN" THEN  
1410     PRINT "Call handover failed. New channel assignment =";Traf_chan  
1420     PRINT "Program terminated."  
1430     STOP  
1440   END IF  
1450   FOR Ms_pwr_lvl=5 TO 15 STEP 5         ! Test power levels 5, 10 & 15.  
1460     OUTPUT Test_set;"CALL:MS:TXL:SEQ ";Ms_pwr_lvl ! Use :SEQ to force  
1470                                       ! sequential execution of  
1480                                       ! the TXLevel command.  
1490   !  
1500 !*****  
1510 ! Step 6: Make Measurements  
1520 !*****  
1530 !  
1540 ! Step 6a: Start a set of concurrent measurements:  
1550 !  
1560   OUTPUT Test_set;"INIT:TXP;PFER;ORFS"  
1570 !
```

```

1580 ! Step 6b: Determine if a measurement is done:
1590 !
1600     LOOP
1610         OUTPUT Test_set;"INIT:DONE?"
1620         ENTER Test_set;Meas_done$
1630 !
1640 ! Step 6c: Obtain measurement results: Each measurement illustrates a
1650 !         different way of reading in results. There is no one right way. The
1660 !         method used is application dependent. Note that the examples do not
1670 !         show all possible ways.
1680 !
1690     SELECT Meas_done$
1700 !
1710     CASE "TXP" ! TX Power measurement done.
1720         OUTPUT Test_set;"FETC:TXP:INT?:POW:ALL?"
1730         ENTER Test_set;Integrity,Txpower(*)
1740         IF (Integrity=0) THEN ! Always check integrity value.
1750             PRINT "TX Power results: TCH =" ;Traf_chan;"and TXL =" ;Ms_pwr_lvl
1760             PRINT USING "5X,""Minimum:""",M2D.2D,"" dBm""";Txpower(1)
1770             PRINT USING "5X,""Maximum:""",M2D.2D,"" dBm""";Txpower(2)
1780             PRINT USING "5X,""Average:""",M2D.2D,"" dBm""";Txpower(3)
1790             PRINT USING "5X,""Std Dev:""",M2D.2D,"" dB""";Txpower(4)
1800         ELSE
1810             GOSUB Bad_measurement
1820         END IF
1830 !
1840     CASE "PFER" ! Phase & Frequency Error measurement done.
1850         OUTPUT Test_set;"FETC:PFER:ALL?"
1860         ENTER Test_set;Integrity,Rms_phas_err,Peak_phas_err,Worst_freq_err
1870         IF (Integrity=0) THEN
1880             PRINT "PFERror results: TCH =" ;Traf_chan;"and TXL =" ;Ms_pwr_lvl
1890             PRINT USING "5X,""RMS Phase Error:""",M2D.2D,"" deg""";Rms_phas_err
1900             PRINT USING "5X,""Peak Phase Error:""",M2D.2D,"" deg""";Peak_phas_err
1910             PRINT USING "5X,""Worst Freq Error:""",M3D.2D,"" Hz""";Worst_freq_err
1920         ELSE
1930             GOSUB Bad_measurement
1940         END IF
1950 !
1960     CASE "ORFS" ! ORFS measurement done.
1970     !
1980     ! This code illustrates a more 'generic' approach to reading measurement
1990     ! results. By using the capabilities designed into high-level
2000     ! measurements, routines that access measurement results do not have to
2010     ! explicitly know what the measurement execution conditions were. That
2020     ! information can be determined at the time the measurement results are
2030     ! queried.
2040     !
2050     OUTPUT Test_set;"FETC:ORFS:INT?" ! Check measurement integrity.
2060     ENTER Test_set;Integrity
2070     IF (Integrity=0) THEN
2080         OUTPUT Test_set;"SET:ORFS:SWIT:FREQ:POIN?" ! Get number of offsets
2090         ! tested.
2100     ENTER Test_set;Points

```

## Comprehensive Program Example

```
2110         IF Points THEN ! Only query if one or more offsets tested.
2120             ALLOCATE Orfs_swit_res(Points),Orfs_swit_offs(Points)
2130             OUTPUT Test_set;"SET:ORFS:SWIT:FREQ?" ! Get measurement offsets.
2140             ENTER Test_set;Orfs_swit_offs(*)
2150             OUTPUT Test_set;"FETC:ORFS:POW?;FETC:ORFS:SWIT?" ! Get results.
2160             ENTER Test_set;Tx_power,Orfs_swit_res(*)
2170             PRINT "ORFS Swit Results: TCH =" ;Traf_chan;"and TXL =" ;Ms_pwr_lvl
2180             PRINT USING "19X,""TX Power ="",M2D.2D,"" dBm""";Tx_power
2190             PRINT "      Offset(kHz)          Level(dBm)"
2200             PRINT "      -----          -----"
2210 Orfs_image: IMAGE 6X,M4D.2D,12X,M4D.2D
2220             FOR J=1 TO Points
2230                 PRINT USING Orfs_image;(Orfs_swit_offs(J)/1.E+3),Orfs_swit_res(J)
2240             NEXT J
2250             DEALLOCATE Orfs_swit_res(*),Orfs_swit_offs(*)
2260         END IF
2270         OUTPUT Test_set;"SET:ORFS:MOD:FREQ:POIN?" ! Get number of offsets
2280                                     ! tested.
2290         ENTER Test_set;Points
2300         IF Points THEN ! Only query if one or more offsets tested.
2310             ALLOCATE Orfs_mod_res(Points),Orfs_mod_offs(Points)
2320             OUTPUT Test_set;"SET:ORFS:MOD:FREQ?" ! Get measurement offsets.
2330             ENTER Test_set;Orfs_mod_offs(*)
2340             OUTPUT Test_set;"FETC:ORFS:POW?;FETC:ORFS:MOD?" ! Get results.
2350             ENTER Test_set;Tx_power,Pwr_30khz,Orfs_mod_res(*)
2360             PRINT "ORFS Mod Results: TCH =" ;Traf_chan;"and TXL =" ;Ms_pwr_lvl
2370             PRINT USING "18X,""30 KHz BW Power ="",M2D.2D,"" dBm""";Pwr_30khz
2380             PRINT "      Offset(kHz)          Level(dB)"
2390             PRINT "      -----          -----"
2400             FOR J=1 TO Points
2410                 PRINT USING Orfs_image;(Orfs_mod_offs(J)/1.E+3),Orfs_mod_res(J)
2420             NEXT J
2430             DEALLOCATE Orfs_mod_res(*),Orfs_mod_offs(*)
2440         END IF
2450     ELSE
2460         GOSUB Bad_measurement
2470     END IF
2480 END SELECT
2490 EXIT IF Meas_done$="NONE"
2500 END LOOP ! If 'WAIT' is returned from 'INIT:DONE?' query, it just falls
2510         ! through the loop.
2520 NEXT Ms_pwr_lvl
2530 !
2540 !*****
2550 ! Step 7: Perform an Intra-cell Handover
2560 !*****
2570 !
2580 NEXT Traf_chan ! The handover is performed at the top of the FOR loop at line
2590                 ! 1300
2600 !
2610 !*****
2620 ! Step 8: Disconnect the Mobile Station From the BSE
2630 !*****
```

```

2640 !
2650 OUTPUT Test_set;"CALL:END"
2660 OUTPUT Test_set;"CALL:CONN:STAT?"
2670 ENTER Test_set;Call_connected
2680 IF Call_connected THEN
2690     BEEP
2700     PRINT "Unable to complete BS termination. Program terminated."
2710     STOP
2720 END IF
2730 PRINT "Program completed."
2740 STOP
2750 !
2760 Bad_measurement: !
2770 PRINT "Measurement error: "&Meas_done$
2780 PRINT "Measurement Integrity value ="&Integrity
2790 RETURN
2800 !
2810 END ! End of program
2820 !
2830 SUB Chk_err_msg_que
2840     COM /Address/ Test_set
2850     DIM Error_message$(255)
2860     Error_flag=0
2870     LOOP
2880         OUTPUT Test_set;"SYST:ERR?"
2890         ENTER Test_set;Error_number,Error_message$
2900         EXIT IF Error_number=0
2910         IF Error_number=-350 THEN
2920             Error_flag=1
2930             PRINT "Error Message Queue overflow. Error messages have been lost."
2940         ELSE
2950             Error_flag=1
2960             PRINT Error_number,Error_message$
2970         END IF
2980     END LOOP
2990     IF NOT Error_flag THEN
3000         PRINT "No errors in Error Message Queue."
3010         SUBEXIT
3020     END IF
3030     STOP
3040 SUBEND

```

## Example Program Without Comments

The following program is basically the same as the example program presented in ["Example Program With Comments" on page 64](#) but comments have been removed and the coding reflects the use of compound commands and complex commands to achieve coding efficiency.

```

10 ! Prog Name: sim_man_ex.txt      Rev: A.0.2      Date Code: 12/18/98
20 OPTION BASE 1
30 COM /Address/ Test_set
40 DIM Swit_offs$(255),Mod_offs$(255)
50 REAL Txpower(4)

```

## Comprehensive Program Example

```
60 Test_set=714
70 PRINTER IS CRT
80 CLEAR SCREEN
90 OUTPUT Test_set;"*RST;SYST:COMM:GPIB:DEB:STAT ON"
100 CALL Chk_err_msg_que
110 OUTPUT Test_set;"CALL:OPER:MODE CELL;;SYST:CORR:SGA -6"
120 OUTPUT Test_set;"CALL:CELL:BAND PGSM;BCH 20;POW:SAMP -85;;CALL:TCH:ARFC 45;TSL 4"
130 OUTPUT Test_set;"CALL:CELL:ACT OFF;MCC 1;LAC 1;MNC 1;NCC 1;BCC 5;ACT ON"
140 OUTPUT Test_set;"SET:ORFS:SWIT:COUN 5;;SET:ORFS:MOD:COUN 10"
150 OUTPUT Test_set;"SET:ORFS:CONT OFF;TIM 60;TRIG:SOUR AUTO"
160 Swit_offs$="400KHZ,-400KHZ,600KHZ,-600KHZ,1200KHZ,-1200KHZ,1800KHZ,-1800KHZ"
170 Mod_offs$=".2MHZ,-.2MHZ,.4MHZ,-.4MHZ,.6MHZ,-.6MHZ,.8MHZ,-.8MHZ,1MHZ,-1MHZ"
180 OUTPUT Test_set;"SET:ORFS:SWIT:FREQ "&Swit_offs$&";:SET:ORFS:MOD:FREQ "&Mod_offs$
190 OUTPUT Test_set;"SET:TXP:COUN 3;CONT OFF;TIM 20;TRIG:SOUR RISE;QUAL ON"
200 OUTPUT Test_set;"SET:PFER:COUN 8;CONT OFF;TIM 30;BSYN MID;TRIG:SOUR PROT;QUAL ON"
210 OUTPUT Test_set;"CALL:PAG:REP OFF;IMSI `001012345678901`"
220 Tries=1
230 LOOP
240 OUTPUT Test_set;"CALL:ORIG;CONN:STAT?"
250 ENTER Test_set;Call_connected
260 EXIT IF Call_connected
270 OUTPUT Test_set;"CALL:END"
280 IF Tries=50 THEN
290 BEEP
300 DISP ""
310 PRINT "Call did not connect after";Tries;". Program terminated."
320 STOP
330 END IF
340 DISP "Call has not connected after";Tries;"attempts. Trying again."
350 Tries=Tries+1
360 END LOOP
370 DISP ""
380 OUTPUT Test_set;"CALL:MS:DTX OFF"
390 FOR Traf_chan=120 TO 124 STEP 2
400 OUTPUT Test_set;"CALL:TCH:SEQ ";Traf_chan;";:CALL:STAT:STAT?"
410 ENTER Test_set;Call_status$
420 IF Call_status$<>"CONN" THEN
430 PRINT "Call handover failed. New channel assignment =";Traf_chan
440 PRINT "Program terminated."
450 STOP
460 END IF
470 FOR Ms_pwr_lvl=5 TO 15 STEP 5
480 OUTPUT Test_set;"CALL:MS:TXL:SEQ ";Ms_pwr_lvl;";:INIT:TXP;PFER;ORFS"
490 LOOP
500 OUTPUT Test_set;"INIT:DONE?"
510 ENTER Test_set;Meas_done$
520 SELECT Meas_done$
530 CASE "TXP"
540 OUTPUT Test_set;"FETC:TXP:INT?;POW:ALL?"
550 ENTER Test_set;Integrity,Txpower(*)
560 IF (Integrity=0) THEN
570 PRINT "TX Power results: TCH =";Traf_chan;"and TXL =";Ms_pwr_lvl
580 PRINT USING "5X,""Minimum:"" ,M2D.2D,"" dBm""";Txpower(1)
```

```

590         PRINT USING "5X,""Maximum:""",M2D.2D,"" dBm""";Txpower(2)
600         PRINT USING "5X,""Average:""",M2D.2D,"" dBm""";Txpower(3)
610         PRINT USING "5X,""Std Dev:""",M2D.2D,"" dB""";Txpower(4)
620     ELSE
630         GOSUB Bad_measurement
640     END IF
650 CASE "PFER"
660     OUTPUT Test_set;"FETC:PFER:ALL?"
670     ENTER Test_set;Integrity,Rms_phas_err,Peak_phas_err,Worst_freq_err
680     IF (Integrity=0) THEN
690         PRINT "PFERror results: TCH =";Traf_chan;"and TXL =";Ms_pwr_lvl
700         PRINT USING "5X,""RMS Phase Error:""",M2D.2D,"" deg""";Rms_phas_err
710         PRINT USING "5X,""Peak Phase Error:""",M2D.2D,"" deg""";Peak_phas_err
720         PRINT USING "5X,""Worst Freq Error:""",M3D.2D,"" Hz""";Worst_freq_err
730     ELSE
740         GOSUB Bad_measurement
750     END IF
760 CASE "ORFS"
770     OUTPUT Test_set;"FETC:ORFS:INT?"
780     ENTER Test_set;Integrity
790     IF (Integrity=0) THEN
800         OUTPUT Test_set;"SET:ORFS:SWIT:FREQ:POIN?"
810         ENTER Test_set;Points
820         IF Points THEN
830             ALLOCATE Orfs_swit_res(Points),Orfs_swit_offs(Points)
840             OUTPUT Test_set;"SET:ORFS:SWIT:FREQ?;:FETC:ORFS:POW?;:FETC:ORFS:SWIT?"
850             ENTER Test_set;Orfs_swit_offs(*),Tx_power,Orfs_swit_res(*)
860             PRINT "ORFS Swit Results: TCH =";Traf_chan;"and TXL =";Ms_pwr_lvl
870             PRINT USING "19X,""TX Power =""",M2D.2D,"" dBm""";Tx_power
880             PRINT "      Offset(kHz)          Level(dBm)"
890             PRINT "      -----          -----"
900 Orfs_image:  IMAGE 6X,M4D.2D,12X,M4D.2D
910             FOR J=1 TO Points
920                 PRINT USING Orfs_image;(Orfs_swit_offs(J)/1.E+3),Orfs_swit_res(J)
930             NEXT J
940             DEALLOCATE Orfs_swit_res(*),Orfs_swit_offs(*)
950         END IF
960         OUTPUT Test_set;"SET:ORFS:MOD:FREQ:POIN?"
970         ENTER Test_set;Points
980         IF Points THEN
990             ALLOCATE Orfs_mod_res(Points),Orfs_mod_offs(Points)
1000            OUTPUT Test_set;"SET:ORFS:MOD:FREQ?;:FETC:ORFS:POW?;:FETC:ORFS:MOD?"
1010            ENTER Test_set;Orfs_mod_offs(*),Tx_power,Pwr_30khz,Orfs_mod_res(*)
1020            PRINT "ORFS Mod Results: TCH =";Traf_chan;"and TXL =";Ms_pwr_lvl
1030            PRINT USING "18X,""30 KHz BW Power =""",M2D.2D,"" dBm""";Pwr_30khz
1040            PRINT "      Offset(kHz)          Level(dB)"
1050            PRINT "      -----          -----"
1060            FOR J=1 TO Points
1070                PRINT USING Orfs_image;(Orfs_mod_offs(J)/1.E+3),Orfs_mod_res(J)
1080            NEXT J
1090            DEALLOCATE Orfs_mod_res(*),Orfs_mod_offs(*)
1100        END IF
1110    ELSE

```

## Comprehensive Program Example

```
1120             GOSUB Bad_measurement
1130             END IF
1140             END SELECT
1150             EXIT IF Meas_done$="NONE"
1160             END LOOP
1170         NEXT Ms_pwr_lvl
1180     NEXT Traf_chan
1190     OUTPUT Test_set;"CALL:END;CONN:STAT?"
1200     ENTER Test_set;Call_connected
1210     IF Call_connected THEN
1220         BEEP
1230         PRINT "Unable to complete BS termination. Program terminated."
1240         STOP
1250     END IF
1260     PRINT "Program completed."
1270     STOP
1280     !
1290 Bad_measurement: !
1300     PRINT "Measurement error: "&Meas_done$
1310     PRINT "Measurement Integrity value ="&Integrity
1320     RETURN
1330     !
1340     END
1350     !
1360     SUB Chk_err_msg_que
1370         COM /Address/ Test_set
1380         DIM Error_message$(255)
1390         Error_flag=0
1400         LOOP
1410             OUTPUT Test_set;"SYST:ERR?"
1420             ENTER Test_set;Error_number,Error_message$
1430             EXIT IF Error_number=0
1440             IF Error_number=-350 THEN
1450                 Error_flag=1
1460                 PRINT "Error Message Queue overflow. Error messages have been lost."
1470             ELSE
1480                 Error_flag=1
1490                 PRINT Error_number,Error_message$
1500             END IF
1510         END LOOP
1520         IF NOT Error_flag THEN
1530             PRINT "No errors in Error Message Queue."
1540             SUBEXIT
1550         END IF
1560         STOP
1570     SUBEND
```