User's Guide

Agilent PDQ-WLR™ revision 2.01

Agilent Technologies

E3185-90000

January 2000

Legal Notice

- **Notice**

  The information contained in this document is subject to change without notice.

  Agilent Technologies makes no warranty of any kind with regard to this material, including but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Agilent Technologies shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance, or use of this material.

- **Warranty**

  A copy of the specific warranty terms applicable to your Agilent Technologies product and replacement parts can be obtained from your local Sales and Service Office.

Other Legal Notices

The executable graphics file is a Korn-shell script which invokes the University of California at Berkeley X11 utility, xgraph, written by David Harrison.

Lotus 123 Release 5 for Windows © Lotus Development Corporation

Microsoft Excel © Microsoft Corporation.

Printing History

Edition	Date	
First	October, 1998	for Revision 1.00
Second	April, 1999	for Revision 2.00
Third	January 2000	for Revision 2.01

1. Installation

2. Software Architecture

6. Hot Carriers and Device Reliability

1 Installation

Requirements Checklist

The following are required before installing (or updating) this product:

Hardware Requirements

- HP Series 745 workstation
- CD-ROM Drive to load software
- 8 MB free disc space

System Software Requirements

Note: The following software must be installed and configured before loading.

Platform	UP-UX	HP-BASIC
S745	10.20	8.02

Test Shell Software

PDQ-WLR is designed to work with Agilent SPECS 2.3 or greater, or as a library of subprograms which may be called from site-specific customized software written in HP BASIC. No kernel, shell, or operating system changes are required to use PDQ-WLR successfully as a stand-alone product or with either test shell.

Parametric Testers

PDQ-WLR is designed to work with the ***Agilent 4062UX*** and ***Agilent 4070 series*** Semiconductor Parametric Testers.

Installing from the Distribution Media

Media Types

PDQ-WLR is distributed on CD-ROM.

Formats

The distribution media are available in ***HP-UX 10.20/swinstall*** format.

Product Identification

The installation product identifier is ***PDQ_WLR***.

Loading the Distribution

- Login to the system as *root.*
- Ensure that neither HP BASIC processes nor Agilent SPECS is running before beginning the installation.
- Performing a backup of the target system is strongly recommended before proceeding with any new software installation or upgrade.
- Select an open *dtterm* window.
- Insert the media into the appropriate drive on your HP workstation.
- Ensure that the /SD_CDROM mountpoint exists, by typing:
  ls -d /SD_CDROM.
  If it does not exist, create it by typing:
  mkdir /SD_CDROM
- Mount the distribution:
  mount -r -F cdfs /dev/dsk/<device_name> /SD_CDROM
- In the *dtterm* window, enter the command ***swinstall*** to bring up the graphical user interface for the installation process.
- ***swinstall*** will automatically present a selected window called *Specify Source.*
- The first selection at the top of the window is labeled **Source Depot Type:** and next to it is a selection button. Use a single left-button mouse click on the selection button to present the depot type menu.

- Select **Local Tape**.
- Type */SD_CDROM/PDQ_WLR.UPD;1* as the **Source Depot Path...**
- Click on the **OK** button to complete the source selection.
- Note that there is one line with the product named ***PDQ_WLR*** in the *SD Install - Software Selection* window.
- Select the line with a single left-button mouse click. The line will be highlighted to indicate that the product is selected.
- Select the *Action* menu from the menubar, and choose *Install (Analysis)*.
- The *Install Analysis* window is presented.
- Click on the **Disk Space...** and **Logfile...** buttons to bring up windows to verify that the target filesystems have sufficient capacity and that the Logfile shows no errors or warnings. Close the *Disk Space* and *Logfile* windows when finished reviewing the data.
- Click on the **OK** button in the *Install Analysis* window to proceed; or on the **Cancel** button to stop if errors or warnings appeared.
- If you had previously installed a patch to PDQ-WLR, taking you to revision 1.01, the Analysis Phase will generate an ERROR that you are attempting to load an earlier version of the PDQ-WLR software. (This is caused by an error in the core revision identifier in the patch.) To proceed, select the *Options* menu from the menubar, and choose *Change Options...* Click on the **Allow installation of lower version than current** option, and close the options window. Repeat the Analysis Phase; you should expect to see a **Downdate WARNING**. If there are no other warnings or errors, you can safely proceed with the installation.
- If installation analysis was successful (you selected **OK**), press the **Yes** button in the *Confirmation* window.
- While the installation proceeds, select the **Logfile...** button in the *Install Window* to monitor the progress of the installation. Drag the *Logfile* window if needed so that the *Logfile* window and the *Install Window* are visible at the same time.
- The installation is finished when the **Done** button in the *Install Window* is selectable. Confirm that there are no errors or warnings in the *Logfile* window.
- Click the **Done** button in the *Install Window*.

- Select the *File* menu from the menubar of the *SD Install - Software Selection* window, and choose *Exit*.
- Unmount the distribution media by typing: *umount /SD_CDROM*.
- The installation is complete. Proceed to System Configuration, the next section.

System Configuration

Defining PDQ-WLR for Your System

PDQ-WLR is defined in two parts for your system:

- Specify the test equipment that PDQ-WLR should recognize and use.
- Specify the run-time control and output characteristics for PDQ-WLR

PDQ-WLR Configuration Methods

PDQ-WLR accomplishes the system configuration using three methods:

- By binding in specific instrument libraries to support your test equipment.
- By automatically generating software during the build process. (Details for this are covered in the next section, **Building PDQ-WLR.**)
- By reading user-defined PDQ-WLR server flag variables. (The flag variables control how PDQ-WLR behaves at run-time.)

PDQ-WLR Configuration Files

There are two PDQ-WLR configuration files, both of which are located in /opt/WLR/etc:

- WLR.config - Defines equipment libraries to bind in, and consequently, also defines what software will be auto-generated. This file is read only by makePDQ-WLRLib and makewlrprog.
- algFlags - Defines the state of the PDQ-WLR flag variables; flags are either *on* or *off*. This file is read only at run-time.

The files are plain text files which must be edited before building PDQ-WLR. The easiest way to edit either of these files is to locate them with the HP-UX file manager tool, and double-click on file to edit.

Configuration Files and Reinstalling PDQ-WLR

Should the need ever arise to reinstall PDQ-WLR, your configuration files ***will not be overwritten*** with the distributed default files. The PDQ-WLR installation process will always attempt to protect work you've already completed.

Upgrading from SWORD or Previous PDQ-WLR Versions

If you were a previous SWORD user, then you may already have similar WLR.config and algFlags files in the /usr/pcs/WLR/etc area on your system. Previous PDQ-WLR users will already have these files in the /opt/WLR/etc directory. Past versions of PDQ-WLR would merge your old WLR.config definition with the new one. Beginning with PDQ-WLR 2.01 this is no longer possible. If you are a previous SWORD user, then you may review your old WLR.config file in /usr/pcs/WLR/etc. If you are a previous PDQ-WLR user, then your old files will be renamed to WLR.config.obsolete and algFlags.obsolete, respectively. If your test environment has not changed then you may wish to review your old files as an aid in editting the /opt/WLR/etc/WLR.config file.

The algFlags BreakPt Flag

In the previous versions of the /opt/WLR/etc/algFlags file, the default for the "BreakPt" flag was enabled. In the new PDQ-WLR the default for this flag is disabled by popular user request. The "BreakPt" should really be set on only if you are developing custom algorithms using the PDQ-WLR error-trapping mechanisms as a template. Normal users, especially those performing production tests, will not want the system to halt on control errors. Control errors will occur normally if you have poor probe pin contact or if there is an error in your test structure layout.

The WLR.config File

The WLR.config file, located in /opt/WLR/etc, specifies the test equipment and software that PDQ-WLR may use.

This file is organized into the follow sections:

- Configuration Overview
- Parametric Tester Configuration Definition
- External Hardware Configuration Definition
- External Software Configuration Definition
- PDQ-WLR Printer Configuration Definition

Configuration Overview

The first section of the file provides a roadmap of the file, its internal variables, and the configuration steps to follow. This section provides essential information for anyone developing software to add to PDQ-WLR.

Parametric Tester Configuration Definition

This second section of the file allows the user to select whether PDQ-WLR will be configured for the Agilent 4070 series or the Agilent 4062UX parametric tester. In addition, it defines what compliment of SMUs are available, and if external equipment is being used, it defines the Aux Port connections.

External Hardware Configuration Definition

Once PDQ-WLR knows, from the previous section, that some external hardware is connected to your parametric tester, you will need to supply a few other critical details. PDQ-WLR needs to know the instrument name, from which it will form a library name to look for. It needs to know the equipment class in order to make the instrument available to the algorithms, and it needs to know the GPIB address for synchronizing command and control of the unit with the parametric tester.

External Software Configuration Definition

Once all of the hardware has been defined, PDQ-WLR will need to bind in a database interface and a prober interface for CV test users requiring full automatic prober control. (Prober control is usually left to the test shell, such as Agilent SPECS; however, the CV tests may require control of the prober in mid-test. PDQ-WLR does not perform direct prober control; it uses a special interface in order to use the test shell's prober controller.)

PDQ-WLR comes complete with Agilent SPECS database and prober interfaces, and with general-purpose interface templates for non-Agilent SPECS users.

In addition to the supplied interface libraries, users are free to develop test software libraries in HP BASIC, and bind these libraries into PDQ-WLR in support of on-site custom algorithm development. If this describes your site needs, then you may also add entries for PDQ-WLR to bind in your software libraries in this section.

PDQ-WLR Printer Configuration Definition

The user has the option of enabling an automatic debug-printout of PDQ-WLR algorithm input and output parameters. In order for this option to work, PDQ-WLR needs to know how to print from your parametric tester workstation controller. This section provides that definition, which is supplied with a default of "| lp", which directs printed output to your print spooler.

Another use of this feature is to print to a log file. This is accomplished by the definition, "| cat >> /tmp/pdqwlr_test.log". The logfile could then be reviewed, while the tests are running, real-time, in a scrollable *dtterm* window, with the command, "tail -f /tmp/pdqwlr_test.log". Note that if you adopt this technique as shown, the logfile will continue to grow in size in the /tmp area until you take steps to shorten or remove it manually.

Whenever you add equipment to your WLR test suite, you will need to update the WLR.config file and rebuild PDQ-WLR.

The WLR.config file is self-documenting. The installed default version is presented here, with its embedded instructions.

```
#
# WLR Source Version: $Revision: 1.18 $
# Last revised date:  $Date: 2000/02/10 21:32:45 $
# Source module name: $RCSfile: WLR.config,v $
#
# (c) Copyright 1997-2000 Sandia Technologies, Inc.  All rights reserved.
# Other copyrights may be listed below.
#
# (c) Copyright 1996-2000, Agilent Technologies, all rights reserved.

#
# Organization:
#   This file is organized into the follow sections:
#     1.  Configuration Overview
#     2.  Parametric Tester Configuration Definition
#     3.  External Hardware Configuration Definition
#     4.  External Software Configuration Definition
#     5.  PDQ-WLR Printer Configuration Definition
#
#   For best results, first print this file.  This may help provide a
#   complete overview of the configuration steps.
#
#   Edit each of the definition sections; you may find it easiest to
#   edit them in the order in which they are presented.
#
#
#/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\
#
#                    1. CONFIGURATION OVERVIEW
#
#\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/
#
#
#
# An entry in this file (WLR.config) has one of the following formats:
#
# CONST   CName      CValue            [# Comment]
# INSTR   Module     Class    Address  [# Comment]
# BCOM    FileName                     [# Comment]
# BLIB    FileName                     [# Comment]
# BPRT    PRINTER IS site-specification
#
# CONST  - Constants used in algorithms
# INSTR  - Instrument driver
# BCOM   - BASIC subprogram of COM definitions to loadsub (loaded first)
# BLIB   - BASIC library file to loadsub
# BPRT   - BASIC definition of site printer
#
#
# -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -
#
# The CONST entries are used to define your parametric tester configuration.
# In addition, if you write your own software to add to PDQ-WLR, you may
```

```
# use this facility to add in any constants of your own choosing.  To
# ensure compatibility with all future PDQ-WLR releases, we suggest that
# all of your constants take the form of EX_<your-constant-name>; the
# "EX_" prefix is reserved for use external to PDQ-WLR.
#
# CONST Parameters=
# -----------------
# CName  : Name of the constant (alphanumeric, 16 characters max)
#          The name is forced to upper case.
#
# CValue : String value to assign (alphanumeric, 16 characters max)
#
# EXAMPLE - How to retrieve a value from the PDQ-WLR constant table:
#   PDQ-WLR provides a BASIC string function called FNWlrconst$ for
#   this.  The function takes the CName argument, and returns the
#   CValue.  If your constant is numeric, you can retrieve and convert
#   it in one call, using VAL(FNWlrconst$("<your-CName>")).
#
# There is no limit to the size of the PDQ-WLR constant table.
#
#
# -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -
#
# The INSTR entries are used to define device drivers for hardware
# which is separate from, or optional to, your parametric tester.
# (This includes your LCR meter, such as the Agilent 4284A.  Your TIS
# software controls hardware at a different level than does PDQ-WLR,
# so your Agilent 4284A address must be re-specified here.)
#
# INSTR Parameters=
# -----------------
# Module : Binary (PROG) module to link in
#          o  Instrument drivers have the format WLR<Module>,
#             for example, the Agilent 8110A driver is in WLRHP8110A.
#          o  Instrument drivers MUST be located in /opt/WLR/lib
#
# Class  : Instrument class. There should be only one entry per class.
#          If you attempt to install more than one instrument of the
#          same class, the PDQ library builder scripts will use the
#          first one found, and print error messages for references
#          to the duplicates.  Here are the classes known to PDQ-WLR:
#          PGU    - pulse generator
#          FREQ   - frequency counter
#          TCHUCK - thermochuck
#          CMU84  - LCR meter
#          QSU    - pA meter
#
#       o  EXAMPLE 1 - REPLACING A PDQ-WLR LIBRARY WITH ONE OF YOUR OWN:
#          If you need to include your own thermochuck in place of the
#          one provided, you may use the /opt/WLR/lib/WLRTEMP314B
#          driver as a template, and call your driver:
#          /opt/WLR/lib/WLR<your_name_here>.  As long as it is of
#          class TCHUCK, the library builder scripts will bind it in
#          automatically for use by PDQ-WLR.
#
#      o  EXAMPLE 2 - ADDING A COMPLETELY NEW TYPE OF INSTRUMENT TO PDQ-WLR:
#          If you have an instrument used only by your custom algorithms,
#          which you would like to add to the PDQ-WLR suite, simply
#          invent a new class name for it.  The library builder scripts
#          WILL give you a warning message about the unknown class.
```

```
#          The warning is simply provided to protect you from editing
#          errors, such as a "PUG" class entry when "PGU" was
#          intended.  You may then use the PDQ-WLR software to access
#          your library.  The following example BASIC code will retrieve
#          your instrument's GPIB address, for the new class METER:
#          My_gpib_addr=VAL(FNWlrconst$("METER"))
#          PDQ-WLR will return either the GPIB address entered below,
#          or a zero if the address cannot be found here.
#          (Programming hints:  we use the zero return to halt our
#          algorithms if the address cannot be found; otherwise, we
#          pass the address to the instrument library.  This provides
#          a convenient mechanism to allow for GPIB/instrument changes
#          without having to re-write code to accept new addresses.)
#
# Address: GPIB address of the instrument
#
#
#
# -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -
#
# Blank line and lines starting with a "#" are ignored. After changing
# this file, you need to do one of the following:
#
#  o Using Agilent SPECS, you need to rebuild your PDQ-WLR library using
#    /opt/SPECS/usr/bin/makePDQ-WLRLib
#
#  o If you write your own code in BASIC, you need to re-run makewlrprog
#
#
#
# -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -
#
# IN THE EVENT THAT THIS FILE BECOMES CORRUPTED, A CLEAN (FACTORY DEFAULT)
# BACKUP COPY IS LOCATED AT:  /opt/WLR/newconfig/etc/WLR.config
#
#
#
#/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\
#
#             2. PARAMETRIC TESTER CONFIGURATION DEFINITION
#
#\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/
#
#
# PDQ-WLR is designed to take advantage of any configuration of
# equipment connections to your Agilent 4062UX or Agilent 4070 Series
# testhead.
# This section consists of one definition, two flags, and two tables.
# The tables are mutually exclusive; use one or the other.
#
# DEFINITION - Agilent 4070 Series Users Only
#
# PDQ-WLR dynamically changes your system optimization level in order
# to control the accuracy and speed characteristics of your Agilent
# 4070 Series tester.
# Upon exit, each PDQ-WLR will reset your system to your preferred
# optimization level, which must be specified here.  This definition
# is IGNORED for Agilent 4062UX configurations.  Refer to your Agilent
# 4070 Series User's Guide for details.
```

```
CONST     OPT_LEVEL     2   # 2=factory default optimization level

#
#
# FLAGS - 0=off/no/false   1=on/yes/true
#
# Flag 1: The Medium Power SMU Flag
#   Most installations will have a high power SMU (HPSMU); PDQ-WLR
#   works best when an HPSMU is available.  However, if your tester
#   does not include an HPSMU, PDQ-WLR can be set to override the
#   HPSMU settings and use a medium power SMU (MPSMU) in its place.
#   The default for this flag is 0 (OFF), which means that you DO have
#   an HPSMU.  If you do not, set this flag to 1 (ON).

CONST     USE_MPSMU     0   # 0=HPSMU installed; 1=no HPSMU, use MPSMU

# Flag 2: The 4062 TIS Flag
#   The Agilent 4062UX Test Instruction Set (TIS) library has a different
#   format for some functions from the Agilent 4070 TIS library.
#   Regardless of which table you use, this flag MUST specify which
#   tester, and therefore which library format, to use.
#   DO NOT ATTEMPT TO USE YOUR Agilent 4070 IN THE 4062-Compatible MODE
#   VIA THE PORT MAPPING TABLE!  Due to the way PDQ-WLR extends your
#   system's functionality with external hardware, TIS errors will
#   almost certainly occur if you try to "fool" PDQ-WLR into treating
#   your Agilent 4070 Series Tester as an Agilent 4062UX.
#   The default for this flag is 0 (FALSE), which means that you are
#   using an Agilent 4070 Series tester.  If you have an Agilent 4062UX,
#   set this flag to 1 (TRUE).

CONST     USE_4062      0   # 0=use 4070 TIS; 1=use 4062 TIS

# Port Mapping Tables
#   These tables are pre-defined for the most common tester/external
#   hardware setups in use with PDQ-WLR; however, it is impossible to
#   anticipate every possible tester configuration.  Cable and configure
#   your tester and external hardware in the fashion most convenient to
#   your facility, and then specify your configuration here.
#
#   You will be required specify the following here:
#     - One high power SMU (HPSMU); if you do not have one, specify
#       the port number of a medium power SMU (MPSMU) and set the
#       USE_MPSMU flag, above.
#     - Three MPSMUs; if you use an MPSMU in place of the HPSMU,
#       your system will need four (4) MPSMUs to function correctly
#       with PDQ-WLR.
#     - Optionally, an Aux port connection for your pulse generator
#       unit (PGU) output.  This is used for charge-pumping measurements.
#     - Optionally, a pair of Aux port connections to use with a
#       picoamp meter unit (QSU) for quasi-static CV and mobile ion
#       measurements.  One port is used for forcing, one for sensing.
#       If you have an Agilent 4071A, you have some Aux ports with pairs
#       of Force/Sense BNC connectors.  DO NOT CONNECT BOTH QSU LEADS
#       TO THE FORCE/SENSE BNCs OF ONE AUX PORT.  Like the CMH/CML
#       connections used by your LCR meter, your QSU will require two
#       separate Aux ports to function correctly.
#
#   SMU Port Definitions:
#     1 - For the Agilent 4070 Series, use the /opt/hp4070/bin/hp4070
#         program to access your testhead to see your SMU configuration.
```

```
#     2 - For the Agilent 4062UX, inspect the front panel of your 4142.
#
#   Aux Port Definitions:
#     Match your cabling setup to your testhead or switch matrix.
#     Refer to the "PDQ-WLR Software User's Guide" for external
#     hardware connection details.  The 4071A  and 4062 Aux port
#     definitions match the external connection labels on the
#     testhead or switch matrix.
#
#   Refer to the "Agilent 4062/UX Programming Reference" or the
#   "Agilent 4071A Programming Reference for BASIC Users" for port
#   details.  PDQ-WLR uses the "FNPort" call to get address connections,
#   and the "FNAux" call for 4062 Aux port connections.
#
#   PDQ-WLR uses the system defaults for the ground unit and Agilent 4284A
#   connections (GNDU, CMH, and CML, respectively).  These connections
#   are not required in these tables.
#
# The 4071A is set by default.  To use the 4062 table, comment out
# the 4071A table (using the "#" character in column 1), and uncomment
# the 4062 table.
#
# In either case, set the correct port numbers.  The defaults supplied
# here may, or may not, be correct for your configuration.
#
# IMPORTANT NOTE: If your installation only has three SMUs total, you
#   may use PDQ-WLR, but without all of the features available in the
#   electromigration codes.  If you only have three SMUs, then do not
#   specify extrusion monitor connections in the electromigration
#   routines; leave the default definition for MPSMU_C in place, and
#   everything will function correctly.
#
# DO NOT ASSIGN THE SAME PORT NUMBER TO DIFFERENT SMUs!
#   If you do, then an unrecoverable TIS error will result.
#
# 4071A Port Mapping Table

  CONST     HPSMU     3          # See USE_MPSMU flag, above
  CONST     MPSMU_A   2          # Entries are required for
  CONST     MPSMU_B   4          #  all SMUs, regardless of type
  CONST     MPSMU_C   5

  CONST     AUX_PGU_FORCE     4  # Connect to Force BNC w/ 50ohm->GND
                                 #  terminator
  CONST     AUX_QSU_FORCE     3  # Connect to Force BNC
  CONST     AUX_QSU_MEASURE   1  # Connect 4140B to triax port if the
                                 #  16054A option is used (set to connect
                                 #  the inner braid (Guard) to ground)
                                 #  Otherwise, use triax-to-coax converter,
                                 #  and select the Sense BNC on an UNUSED
                                 #  Aux port - ** OR ** - use two triax-
                                 #  to-coax converters, and run a coax
                                 #  from the 4140B to Aux 1 or 2.

# NOTE- If Aux ports are all used, share ports between PGU, QSU, and
#       LCR functions; swap cables between tests.
#
#
# 4062 Port Mapping Table
```

```
# CONST     HPSMU     2          # See USE_MPSMU flag, above
# CONST     MPSMU_A   1          # Entries are required for
# CONST     MPSMU_B   3          #  all SMUs, regardless of type
# CONST     MPSMU_C   4

# CONST     AUX_PGU_FORCE     3  # Connect to Aux BNC w/ 50ohm->GND
                                 #  terminator
# CONST     AUX_QSU_FORCE     2  # Connect to Aux BNC
# CONST     AUX_QSU_MEASURE   1  # Connect 4140B to triax port using a
                                 #  triax-to-coax converter

# NOTE- If Aux ports are all used, share ports between PGU, QSU, and
#       LCR functions; swap cables between tests.
#
#
#
#/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\
#
#             3. EXTERNAL HARDWARE CONFIGURATION DEFINITION
#
#\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/
#
#
#
# If you have any of the optional supported instruments listed
# below, you must perform TWO steps here:
#  1 - Uncomment the line(s) referring to each instrument selected
#  2 - Replace the sample GPIB select code and address with the
#      the actual value for the instrument.
# For example, if you have an Agilent 8110A at address 25 on your primary
# GPIB bus, remove the "#" character from the HP8110A line, and
# replace our example "720" with "725".
#
# *** NOTE:  Many installations have a thermochuck on the same GPIB
#            as the prober, which is often connected to the secondary
#            GPIB card.  For most HP 745i workstation installations,
#            the secondary GPIB has a select code of "25".  This table
#            shows an example Temptronics thermochuck connected to
#            the secondary GPIB (select code 25) at address 09; for
#            complete address specification of 2509.  Your superuser
#            or Agilent SE can assist you with your select codes by
#            interrogating your local HP-UX hwconfig.cf file.
#
# DO NOT CHANGE YOUR INSTRUMENTS TO MATCH THIS TABLE.
# Change this table to match your instruments.
#
# INSTR   HP8110A        PGU       720    # pulse generator
# INSTR   HP8160A        PGU       720    # pulse generator

# INSTR   HP5316B        FREQ      707    # frequency counter

# INSTR   HP4284A        CMU84     717    # LCR meter

# INSTR   HP4140B        QSU       714    # pA meter

# Thermochucks also require a temperature range to be set

# INSTR   TEMP314B       TCHUCK   2509    # Temptronics thermochuck
# CONST   LOWER_TEMP     1                # Temptronics
```

```
# CONST   UPPER_TEMP     300              # Temptronics

# INSTR   TSK90A_CHUCK   TCHUCK   2509    # TSK 90A thermochuck
# CONST   LOWER_TEMP     50               # TSK 90A
# CONST   UPPER_TEMP     135              # TSK 90A

# INSTR   TELP8_CHUCK    TCHUCK   2509    # TEL P8 thermochuck
# CONST   LOWER_TEMP     50               # TEL P8
# CONST   UPPER_TEMP     150              # TEL P8

# INSTR   EGCHUCK        TCHUCK   2509    # Electroglas/OEM thermochuck
# CONST   LOWER_TEMP     20               # Electroglas
# CONST   UPPER_TEMP     130              # Electroglas

# INSTR   SPECS_CHUCK    TCHUCK   2509    # Use whenever your Agilent SPECS
                                          # prober library already has
                                          # thermochuck control routines.
# CONST   LOWER_TEMP     -1               # SET THIS PER YOUR CHUCK!
# CONST   UPPER_TEMP     -1               # SET THIS PER YOUR CHUCK!

# Set the maximum time for a single hot chuck temperature range,
# in seconds.  The default is 2 hours.

  CONST   MAX_CHUCK_TIME  7200

#
#
#/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\
#
#             4. EXTERNAL SOFTWARE CONFIGURATION DEFINITION
#
#\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/
#
#
# Prober interface and database interface libraries are required.
# The default shows the use of the PDQ-WLR bindings to Agilent SPECS.
#
  BLIB    /opt/WLR/lib/WLR_SPECS_Prober
  BLIB    /opt/WLR/lib/WLR_SPECS_db
#
# BLIB    /opt/WLR/local/WLR_BASIC_Prober
# BLIB    /opt/WLR/local/WLR_BASIC_db
#
# HCI_MOS is now obsolete - refer to your User's Guide for details
# Uncomment one of the following lines. No_HCI_MOS prevents use
# of the HCI_MOS routine, and saves valuable library space.
# Users of HCI_MOS still wishing to execute custom-written
# hot carrier software will need to keep the HCI_MOS_Lib
# default.
#
# The "Save HCI Tables" algFlags entry interacts only with
# HCI_MOS, not HCI_3_MOS.
#
  BLIB    /opt/WLR/lib/HCI_MOS_Lib
# BLIB    /opt/WLR/lib/No_HCI_MOS_Lib
#
# For BASIC users, edit these templates in /opt/WLR/lib:
#   WLR_BASIC_Prober_template
#   WLR_BASIC_db_template
# Store the results in an accessible directory, such as /opt/WLR/local,
```

```
# and specify them INSTEAD of the WLR_SPECS versions above.
# (If you are using the Agilent-supplied EG, TEL, MJC, or TSK libraries,
# the prober template may be used as is, for the Pup and Pdown entries.)
# Prober light on/off functions will have to be provided separately,
# if not available.
#
# Here are examples of using BCOM and BLIB to support customized
# libraries which you may build at your site.  The examples show
# that you've added code to the supplied /opt/WLR/local path,
# but you are really free to refer to files in any location.

# BCOM    /opt/WLR/local/my_basic.com

# BLIB    /opt/WLR/local/my_algorithm_library

#
#
#/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\
#
#              5. PDQ-WLR PRINTER CONFIGURATION DEFINITION
#
#\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/
#
#
#
# Either a direct GPIB device, such as address 701, or your site
# line printer may be specified (via a pipe).  Piped printing may
# include any of the normal printing flags.  Only one BPRT statement
# should be made; the primary printer via lp is the default.
# To access this printer, ensure that the "+Printer On" entry
# in the /opt/WLR/etc/algFlags file is uncommented.  To use this
# feature in your software, add the following code to your algorithms:
#
#    Prt_on=FNWlrx_srvrflag("Printer On")
#    IF Prt_on=1 THEN CALL Wlr_printer     !Send outputs to printer
#    PRINT <your data here>
#    PRINT <more data there>
#    IF Prt_on=1 THEN PRINTER IS CRT

  BPRT   PRINTER IS "| lp"

# BPRT   PRINTER IS 701

# END OF FILE
```

SMU Configurations

The Agilent 4062UX has a single configuration when a high-power SMU is installed. In contrast, the Agilent 4070 series has a wider variety of SMU configurations.

You can determine your Agilent 4070 series SMU configuration by executing /opt/hp4070/bin/hp4070 and inspecting the DC Sources table.

The following table shows the translation from the Agilent 4062UX fixed configuration to the configuration names used in the WLR.config file.

SMU Configuration Number to Name Translation

Agilent 4062UX Configuration	PDQ-WLR SMU Names
SMU 1	MPSMU_A
SMU 2	HPSMU
SMU 3	MPSMU_B
SMU 4	MPSMU_C

The algFlags File

The algFlags file specifies the run-time control and output characteristics for PDQ-WLR.

The /opt/WLR/etc/algFlags file needs to be edited to reflect your operation preferences. We have set some flags (using the "+" sign) which you may find helpful. You may want to enable input and output debugging, and turn the printer on to redirect the debug data to your system printer. The output will be sent to the device or spool specified by the ***BPRT*** variable in the ***WLR.config*** file.

The algFlags file is read whenever you start executing a PDQ-WLR testplan under Agilent SPECS or HP BASIC, or whenever the GET_WLR_FLAG data management algorithm is run. GET_WLR_FLAG allows you to build test plans where, for example, you may pause the test, change the flags, and continue testing with new flag parameters.

The algFlag is completely self-documenting. The installed default version is presented here, with its embedded instructions. Note that the PDQ-WLR server flag processor will first look for a copy of this file in the user's local directory area, then in the /opt/WLR/etc global area.

```
#
# WLR Source Version: $Revision: 1.15 $
# Last revised date:  $Date: 2000/02/02 23:53:24 $
# Source module name: $RCSfile: algFlags,v $
#
# (c) Copyright 1997-2000 Sandia Technologies, Inc.  All rights reserved.
# Other copyrights may be listed below.
#
# (c) Copyright 1996-2000, Agilent Technologies, all rights reserved.

# ------------------------------------------------------------------------
# --  PDQ-WLR Algorithm Flags                                            --
# --                                                                     --
# --  Server flags govern the run-time behavior of PDQ-WLR, and          --
# --    may be thought of as user-definable overrides.                   --
# --    Flags have two states - ON and OFF.  PDQ-WLR will treat          --
# --    missing flags as OFF.                                            --
```

```
# --                                                                    --
# --  Server flags are defined in an algFlags file.                     --
# --                                                                    --
# --  PDQ-WLR will search for the algFlags files as follows:            --
# --    - search the local directory, and use algFlags file if found    --
# --    - search /opt/WLR/etc, and use algFlags file if found           --
# --                                                                    --
# --  Global flag definition, via /opt/WLR/etc/algFlags, is the         --
# --    preferred definition method.                                    --
# --                                                                    --
# --  Blank lines and lines starting with a "#" are ignored             --
# --  A leading "+", without a "#" preceding it, initializes the        --
# --    the flag to ON.                                                 --
# --  For SPECS and PDQ-WLR/BASIC users, "+" turns the flag ON;         --
# --    omission of the "+", commenting out the line with a "#",        --
# --    or removing the flag from the algFlags file, turns the flag     --
# --    OFF.                                                            --
# --                                                                    --
# --  Capitalization and white space will be ignored for flags.         --
# --                                                                    --
# --  There can be a maximum of 16 flags defined. A flag can be 1 to    --
# --  16 characters in length.                                          --
# --                                                                    --
# --                                                                    --
# ------------------------------------------------------------------------

# IN THE EVENT THAT THIS FILE BECOMES CORRUPTED, A CLEAN (FACTORY DEFAULT)
# BACKUP COPY IS LOCATED AT:  /opt/WLR/newconfig/etc/algFlags


# Plot device curve (default is ON)
# Example: This flag allows you to build a testplan with all plotting
# turned on, for development testing, and then for production testing
# without on-screen graphs, you can turn this flag off.  All plotting
# will be disabled without having to modify your testplan.
 +Plot On

# Wait for user to "CLOSE" on-screen xgraph window before continuing
 +Plot Wait

# Display initial values of algorithm parameters
 +Debug Entry
```

```
# Display exit values of algorithm parameters
 +Debug Exit

# Send debug data to WLR printer if on, to the HP BASIC window if off
# +Printer On

# If a trapped system error occurs, pause for user to debug using HP BASIC
# +Breakpt

# For users with automatic probers w/ controllable light option
# Refer to PDQ-WLR documentation for details
# (See WLR_SPECS_Prober or WLR_BASIC_Prober_template in /opt/WLR/lib;
#  refer to the Plight_on_xl and Plight_off_xl subprograms.)
# (Prober Light used for ONLY for CV measurements)
# Users are prompted for manual action if this flag is off
# Action specified in the Plight_on_xl and Plight_off_xl
# routines will be taken if this flag is on
# +Prober Light

# PDQ-WLR performs CV measurements which require the ability to
#  raise and lower the wafer chuck
# SPECS users should indicate if they have configured their test shell
#  to use an autoprober
# BASIC users should set this if they have bound an autoprober driver
#  to the WLR_BASIC_Prober library
# Users are prompted for manual action if this flag is off
# Actions specified in the prober interface library are taken
# if this flag is on
# (Auto Prober used ONLY for CV measurements)
 +Auto Prober

# PDQ-WLR algorithms can produce comma-separated-value (.csv) files
#  for export to spreadsheets.  The following flag determines if the
#  the .csv files will be in DOS or UNIX format.  For Lotus 1-2-3*,
#  the flag should be on; for Microsoft Excel*, the flag may be left off.
#  Files can always be converted later using the HP-UX ux2dos utility.
#  (* Copyrights, Lotus Development Corporation and Microsoft Corporation,
#   respectively.)
# +Csv DOS Format

# The HCI_MOS algorithm can be set by this flag to output its stress
```

```
#  tables in .csv file format.  If this flag is on, the file will always
#  be saved to the /var/opt/WLR/database/data/hci/stress_tables directory;
#  the filename will be "Hci_mos_date/time.csv".  Each file will be
#  pointed to by each Hci_mos database entry.
# This flag interacts with HCI_MOS, not HCI_3_MOS.
# +Save HCI Tables

# END OF FILE
```

The HP-UX Kernel

PDQ-WLR was originally designed to run under Agilent SPECS. HP BASIC users of PDQ-WLR should set the following kernel parameters as is done for Agilent SPECS installations.

Kernel Parameter	Default	Recommendation
maxuprc	50	150
shmmax	0x4000000	0x8000000
msgtql	40	256

To do this, you use the HP-UX SAM utility (`sam`). Refer to "Constructing an HP-UX System" in *System Administration Tasks* HP-UX manual for changing kernel parameters.

Building PDQ-WLR

The Build Process

The build process begins when the installation and configuration definitions are completed.

PDQ-WLR supplies two build scripts:

- makePDQ-WLRLib - for Agilent SPECS users
- makewlrprog - for customized HP BASIC systems developed for use outside of Agilent Technologies test shells

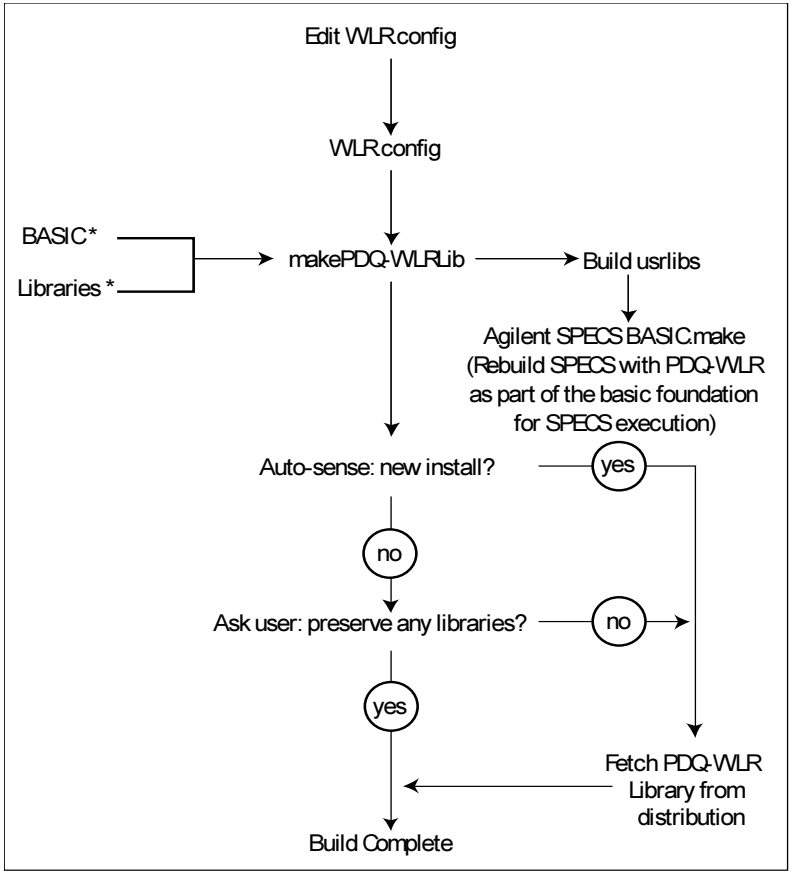
The build process will take the necessary files from /opt/WLR/BASIC and /opt/WLR/lib and copy them into appropriate locations in the software test environment. In the case of Agilent SPECS, two user libraries will be built in /opt/SPECS/usr/lib; for HP BASIC users, the appropriate routines will be bound into a single file and appended to your customized main program.

When to Build PDQ-WLR

The PDQ_WLR build process should be performed whenever a new installation occurs or the /opt/WLR/etc/WLR.config file changes. For HP BASIC users, the build process will have to be performed for each new main program, and, as implied, for each existing main program when the WLR.config file changes. (Remember, the WLR.config file should change whenever your test equipment changes.)

How to Build PDQ-WLR for Agilent SPECS Use

PDQ-WLR/SPECS Build Process



* Note: the BASIC and Library files become part of the SPECS foundation and cannot be edited with AlgBuild; changes must be made with HP BASIC in the /opt/WLR areas.

- Login as ***root***
- Ensure that neither HP BASIC processes nor Agilent SPECS is running before beginning the build process.
- Verify that the WLR.config file is set up for your test system configuration
- Execute /opt/SPECS/usr/bin/makePDQ-WLRLib

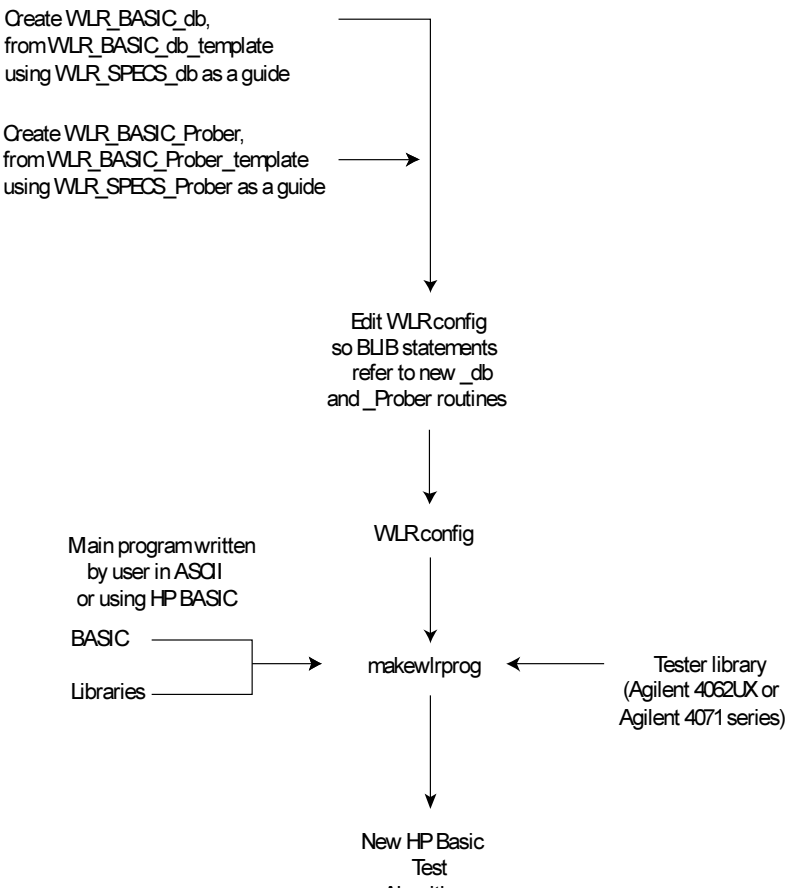
- Review the installed configuration for reported errors

The makePDQ-WLRLib script will report its progress to the screen during execution, and also copies the screen output to the/opt/SPECS/usr/lib/PDQWLR.log file. For subsequent PDQ-WLR builds, the information will be appended to the PDQWLR.log file. Review the contents of this text file for any reported errors. If errors occurred, the build process will stop, and PDQ-WLR not be ready for Agilent SPECS use.

When the build process is successfully completed, PDQ-WLR becomes the default measurement library for Agilent SPECS use. The build script changes the /opt/SPECS/sys/config/sysconf text file ***MEASLIB*** entry to accomplish this; your previous configuration is saved in /opt/SPECS/sys/config/sysconf.bak. Contact your Agilent SPECS system administrator for details regarding the sysconf file.

How to Build PDQ-WLR for HP BASIC Use

PDQ-WLR/BASIC Build Process



- Build a main program in HP BASIC, supplying inputs for the PDQ-WLR algorithms you'll use, and call those algorithms. (The argument list each for each algorithm is described in Chapters 4, 5, and 6.) The main program file may contain any number of customized subprograms and functions in addition to the main program.
- Ensure that neither HP BASIC processes nor Agilent SPECS is running before beginning the build process.
- Verify that the WLR.config file is set up for your test system configuration
- Execute the /opt/WLR/bin/makewlrprog build script.

The makewlrprog syntax is:

/opt/WLR/bin/makewlrprog mainprog *options*

mainprog	the file containing the main program. This can be an HP BASIC LIF file or ASCII file (each line beginning with a statement number). PDQ-WLR will automatically sense the file type and either LOAD or GET the file correctly.

The following options are supported:

–o *outputfile*	Specifies name of the PDQ-WLR program to create. Default is WLR.PROG and the output file is always an HP BASIC LIF type.
–a *algorithm*	Specifies a PDQ-WLR algorithm to attach. The algorithm must be located in the **/opt/WLR/BASIC** directory.
–i *algfile*	Specifies a file that has the names of PDQ-WLR algorithms to attach. Same as doing multiple *–a* options.
–l *proglib*	Specifies a user program library file to LOADSUB in. For user-defined library files which are always included in customized PDQ-WLR programs, it's more convenient to specify these using BLIB statements in Section 4 of the WLR.config file.
–h	Suppress the automatic TIS and PDQ-WLR initialization calls. Normally, there are two initialization calls: for TIS, *Init_system* is called, and for PDQ-WLR, the server flags file (algFlags) is read. Experienced TIS users may explicitly call *Init_system* in their HP BASIC main program; and would use this option to suppress an additional *Init_system* which they might consider out-of-sequence for their needs. For these cases, PDQ-WLR will always sense if server flag initialization is required, and perform it automatically. (SWORD users upgrading to PDQ-WLR will note that this is an improvement in ease of use for makewlrprog.)

-t *tis_id*

Specifies which TIS to bind to the HP BASIC main program. Valid options are *4062*, *4071*, and *4071_prober*; and *no_tis*. For compatibility with existing SWORD installations, the default is *4062*. The Agilent 4070 series parametric tester is supplied with two TIS libraries. Using the *4071* option binds in the /opt/hp4070/lib/TIS4070 library; using the *4071_prober* option binds in the /opt/hp4070/lib/TIS library. Refer to the "Agilent 4071A Semiconductor Parametric Tester Programming Guide for BASIC Users" for details on required libraries for Agilent 4070 series programming. Users requiring Agilent-supplied prober libraries in addition to the appropriate TIS, or who are providing their own custom-designed prober libraries in HP BASIC, should specify these libraries with a BLIB statement in Section 4 of the WLR.config file; a good practice would be to place the new BLIB statements immediately after the prober interface library specification in the WLR.config Section 4 configuration. Ensure that your prober library subprograms are properly referred to by your PDQ-WLR prober interface library. If there is a disagreement between the caller and the callee, it's better to modify the PDQ-WLR prober interface library than it is to change the supplied prober libraries; this is why the prober interface libraries exist as a software layer provided to you in PDQ-WLR. Note that if it is your intention to attempt to support multiple TIS configurations with a single WLR.config file definition, then using the *-l* option of makewlrprog may be a better way to control prober library loading.

The no_tis option is supplied for those users with existing routines that *loadsub* the appropriate *TIS* at runtime.

Preparing Agilent SPECS Frameworks

If you have Agilent SPECS version 2.4 or later, a complete framework has been installed for you use. Refer to ***Agilent PDQ-WLR Using Agilent SPECS***.

If you have Agilent SPECS version 2.3 or earlier, the PDQ-WLR installation process created a sample framework file named /opt/SPECS/usr/fwk/PDQWLR.fwk. This framework is essentially a copy of the Agilent SPECS-supplied System Framework with additional tag variables defined for the electromigration test ***Max_ro*** parameters, and the user-arrays and control variables for hot carrier tests.

If you are familiar with Agilent SPECS frameworks, you may create similar variables in the Tag Spec for your existing, preferred frameworks. Either way, you have two options for preparing a framework for PDQ-WLR use:

- Modify your existing framework which you intend to use with PDQ-WLR to include the new tag variables.
- Modify the PDQWLR.fwk framework to recognize your prober, and use it for PDQ-WLR testing.

If you would like to add new tag variables to your preferred, existing framework, three steps are required:

- Start Agilent SPECS, and use the Outliner tool to access the PDQWLR.fwk file.
- Note the convention used at the end of the "SYSTEM Tag spec" Spec for specifying the Ntime control variable, and the Max_ro, Times, and Pa arrays.
- Use the Outliner tool to access and edit your preferred framework; add the tag variables into your Tag Spec as they appeared in the PDQWLR.fwk file.

To use the PDQWLR.fwk framework, two steps are required:

- Start Agilent SPECS, and use the Outliner tool to access the PDQWLR.fwk file.
- Modify the "System Tag spec" Spec to reflect the correct GPIB address of your prober.

Be sure to specify the correct framework in the Agilent SPECS Test Plan Navigator when conducting PDQ-WLR tests.

Installation Error Recovery

By this point, you may have modified any of the three following files:

- /opt/WLR/etc/WLR.config
- /opt/WLR/etc/algFlags
- /opt/SPECS/usr/fwk/PDQWLR.fwk

In the event that these files became corrupted, and you'd like to start over, the original, distribution default versions are located in:

- /opt/WLR/newconfig/etc/WLR.config
- /opt/WLR/newconfig/etc/algFlags
- /opt/WLR/newconfig/SPECS/fwk

Never edit or otherwise modify any files in the "newconfig" area. Always copy the originals to the correct target directory, and work from there. Of special importance is the /opt/WLR/newconfig/revision file; the build scripts depend on it.

Readme

PDQ-WLR is distributed with an /opt/WLR/newconfig/README file. ***Please consider this file required reading before running any of the PDQ-WLR algorithms.*** It will contain any information which may apply to the latest software release that may not have made the publishing deadline for this manual, and often contains tips for building PDQ-WLR and managing PDQ-WLR files.

An /opt/WLR/newconfig/Special_Notes file is also included and should be considered as required reading.

Recovery from Tester Faults

Every effort has been made to make the PDQ-WLR software suite robust against input errors, unexpected device failures, and tester faults which may occur during WLR testing.

Input Errors

The Agilent Technologies test shells will attempt to trap user input errors, and will succeed in doing so the vast majority of cases. This is done by comparing each input to the range of legal values for that input. HP BASIC users will be happy to note that the PDQ-WLR algorithms repeat this level of legal input checking.

There does exist a class of input errors which cannot be trapped by the shell, such as when a current and voltage are given as legal inputs, but cannot be accommodated by the chosen SMU. For that class of errors, the PDQ-WLR algorithms will generate an input parameter error message and display it on the HP BASIC window. If your test seems to "run instantly" and do nothing, repeat the test with the HP BASIC window open, and look for these error messages. Always refer to the algorithm descriptions, later in this manual, for explanations regarding the synergy of user inputs.

System Halt Initiated by PDQ-WLR

There is a class of test errors which cannot be trapped by the above methods. The best example is setting a stress condition so low that a given algorithm will run forever unless halted. For this class of errors, rather than return to the shell, some of the PDQ-WLR algorithms will halt the system. This is done to prevent you from probing a large number of devices on one or more wafers, only to discover later that the only thing accomplished was a collection of system time-outs, rather than valid data. In this event, PDQ-WLR will halt your HP BASIC system, and produce a new window of suggested recovery instructions and a statement that PDQ-WLR has purposely halted your system. The HP BASIC window will always contain error information for these cases.

System Halt Due to TIS Error

The PDQ-WLR software has been tested extensively during its development and TIS errors should not occur. The only known possible TIS errors that can be generated under PDQ-WLR at this time happen if you input an illegal pad-to-pin translation table for your probecard under Agilent SPECS, or input a non-existent pin directly to a PDQ-WLR routine using either the Agilent SPECS "Exec Algorithm" mode or by calling the routine directly from your own custom HP BASIC routine.

In the event of a TIS error that you have verified was not related to a bad pin number input, please record ***ALL*** input data, the test conditions, and the TIS error message, and contact your Agilent Technologies representative. You may be asked for copies of the WLR.config and algFlags files that were valid for the PDQ-WLR software when the error occurred, as well as a copy of the testplan related to the error. To continue using your system, please do the following:

- Open the HP BASIC window, and in it, enter ***run*** (function key F3).
- Assuming that you are using the Agilent SPECS Navigator tool, select ***Exec Algorithm*** from the ***Execute*** pull-down menu bar.
- In the Algorithm Browse window, select the algorithm to stop the tester. The name of the algorithm will vary depending upon the version of TIS you are using; look for a name such as ***TESTER_STOP*** or ***XXX_STOP*** **(where XXX is your tester name)**. Execute this algorithm. Failure to do so will cause the system to continue to generate meaningless TIS error messages.
- Close the Algorithm window to return to your normal test environment. For Agilent SPECS, you may now edit and run testplans. For HP BASIC, you may need to ***Quit*** your execution window in order to re-start any tests.

If the TIS error occurred after recovering from a system halt initiated by PDQ-WLR, then you may consider the matter to be much less serious, in that no software execution or hardware fault exists per se; however, it would indicate a defect in the recovery instructions that were provided to you. Please share any such occurrences with your Agilent Technologies representative so that corrective action may be taken for the next PDQ-WLR documentation release.

2 Software Architecture

Overview

PDQ-WLR is a software product designed to test wafer level reliability (WLR) test structures using Agilent semiconductor parametric testers for WLR assessments by semiconductor manufacturers and users. PDQ-WLR is in part a derivative, descendent software product developed from the SWORD software developed in turn at Sandia National Laboratories (SNL). The SWORD software is licensed to Sandia Technologies, Inc. (ST) by Sandia National Laboratories.

This chapter describes the software architecture, algorithm features, and compatibility with existing SWORD-based tests.

User Requirements

The WLR user requires two classes of information on the fundamental building blocks of a semiconductor process. These are information on the intrinsic (long term) reliability suitable for extrapolation to normal operating conditions and information on defects or contamination that might occur in the process. Both sets of information are important in developing a new process, qualifying a process for production and release and routinely monitoring the health of a qualified process over time. This information can be obtained through the use of destructive or, in some cases, non-destructive tests. Intrinsic and defect reliability parameters are listed below per fundamental semiconductor building block (interconnect, insulators and transistors).

Building Block	Intrinsic Reliability	Defects
Interconnect	Lifetime Activation Energy Current Density Exponent Grain Structure Statistical Distribution	Void Size Via & Contact Filling Barrier Integrity Step Coverage Alignment Patterning & Etch
Dielectric	Lifetime Electric Field Coefficient Statistical Distribution	Heavy Metal Contamination Silicon Defects Particles and Precipitates Ionic Contamination Plasma Damage Interface States Trapped Charge
Transistor	Lifetime Drain Voltage Coefficient Substrate Current Coefficient Statistical Distribution	Implant Shadowing Interface States Water Outgassing LDD Overlap Alignment Interface Uniformity Hydrogen Contamination Plasma Damage

Requirements Correlation Matrix

The mapping of each algorithm into the user requirements is summarized below. Parametric measurements provide non-destructive information on the process and are used in conjunction with reliability measurements.

Algorithm	Parametrics	Intrinsic Reliability	Defects	Test Standard	Description
BEM_3_EMR	X		X	n/a	Breakdown Energy of Metals
CONST_I_3_EMR			X	n/a	Constant Current EM
ISO_4_EMR	X	X	X	JEDEC JESD 61	Isothermal EM
KELRES_3_EMR	X			JEDEC JESD 33-A	Kelvin Resistance
SWEAT_3_EMR			X	JEDEC JEP 119	SWEAT EM
TCR_3_EMR	X			JEDEC JESD 33-A	Temperature Coefficient Resistance
TVP_3_EMR	X			n/a	Temperature vs. Power
CV_3_CAP	X		X	n/a	Calculate Dit at Vfb, Vmg, and Qsmin
HFCV_3_CAP	X			n/a	High-frequency Capacitance vs. Voltage
JRAMP_3_CAP			X	JEDEC JESD 35	Current Ramp Breakdown
JTDDB_3_CAP		X	X	n/a	Constant Current TDDB
MBLION_3_CAP	X		X	n/a	Mobile Ions
QCV_3_CAP	X			n/a	Quasi-static Capacitance vs. Voltage
VRAMP_3_CAP			X	JEDEC JESD 35	Voltage Ramp Breakdown
VTDDB_3_CAP		X	X	n/a	Constant Voltage TDDB
CPF_3_MOS	X	X	X	ASTM F1340-91	Charge Pumping by Frequency
CPV_3_MOS	X	X	X	n/a	Charge Pumping by Voltage
DELTAL_M2F	X			n/a	Delta Length, Drawn vs. Actual
DOP_MOS	X			n/a	Substrate Doping Concentration

Algorithm	Parametrics	Intrinsic Reliability	Defects	Test Standard	Description
HCI_MOS		X	X	JEDEC JESD 28, 60	Hot Carrier Injection
HCI_3_MOS		X	X	JEDEC JESD 28, 60	Hot Carrier Injection
HCSTRESS_MOS		X	X	JEDEC JESD 28, 60	Hot Carrier Stress
STRSMEAS_2_MOS		X	X	JEDEC JESD 28, 60	Hot Carrier Stress
IDIBIG_MOS	X	X	X	n/a	MOS FET Currents
IDVD_MOS	X	X		n/a	MOSFET Drain I and V vs. Gate V
MAXSTRS_MOS		X		n/a	Maximum Stress
S_MOS	X	X		n/a	Subthreshold Slope
VPT_MOS	X	X		n/a	Voltage Punchthrough
VT_MOS	X	X		JEDEC JESD 28, 60	MOSFET Vt, Gm, and Id
CALC_ID_MOS	X	X		n/a	Calculate Id Ib Ig
CALC_VT_MOS	X	X		n/a	Calculate Vth
LIFETIME_2_MOS		X	X	JEDEC JESD 28, 60	Lifetime Plot
MBLION_SHS		X		JEDEC JESD 28, 60	Mobile Ions

Algorithm Definition and Classes

The term algorithm is generally used here to describe a WLR software code with the following behaviors:

- Inputs are received via the a subprogram invocation from a managing routine; for example, the Agilent SPECS test shell or a user-supplied stand-alone HP BASIC program.
- It accepts user inputs for device under test (DUT) connectivity, and uses an Agilent parametric tester test instruction set (TIS) library to connect the DUT to the tester.
- It accepts user inputs to define test conditions and DUT characteristics.
- It performs necessary error checking on input parameters.
- It uses its inputs for active control of the tester via the TIS, and for WLR-related computations.
- It outputs data in graphical and/or tabular form; determined by user input.
- It is not interactive for normal execution.
- It accepts an override to allow operator interaction during prober-control operations, if applicable; for example, DUT disconnection and light-cycling for CV measurements.
- It accepts an override to allow operator interaction during its graphics-output phase, if applicable.
- It accepts an override to allow operator interaction in the case of a critical system error.
- It may force operator interaction if an illegal, critical condition is detected in mid-test.

There are five classes of PDQ-WLR algorithms:

- Electromigration algorithms, which contain "EMR" in the name for "electromigration routine". Included in this is the SET_TCHUCK routine.
- Hot carrier algorithms, which contain "MOS" in the name, because the DUTs are usually in the form of a metal oxide semiconductor. (Naming exception: DELTAL_M2F, which uses two MOS DUTs simultaneously.)

- Oxide algorithms, which contain "CAP" in the name, because the DUTs are usually in the form of a capacitor. (Exception: MBLION_SHS, which uses a self-heating test structure.)
- Computational algorithms for hot carrier use without DUT inputs; these are LIFETIME_2_MOS, CALC_VT_MOS, and CALC_ID_MOS. This class also contains the HC_LINK_2_MOS routine, which is used to associate lifetime data with stress data, and PD_LINK_MOS to build plasma-damage categories within the PDQ-WLR data area.
- Data management algorithms, such as GET_WLR_FLAG, RECORD_DATA, PLAYBACK_DATA, PDQ_DB_ON, and PDQ_DB_OFF.

Algorithm Characteristics

The following table shows the algorithms' characteristics. For all algorithms, it indicates which perform input error-checking, which use a parametric tester TIS to connect to a device, which produce graphs and comma-separated value (CSV) files, which require equipment in addition to the parametric tester, and which require prober control during testing.

Algorithm	Input Error Check	Connect DUT[1]	Graph	CSV File	Data-base Output	External Equipment Needed[2]	Controls Prober[3]
BEM_3_EMR	X	X	X	X	X		
CONST_I_3_EMR	X	X	X	X	X		
ISO_4_EMR	X	X	X	X	X		
KELRES_3_EMR	X	X			X		
SWEAT_3_EMR	X	X	X	X	X		
TCR_3_EMR	X	X	X	X	X	TCHUCK	(Note 10)
TVP_3_EMR	X	X	X	X	X		
TVP_3_EMR						TCHUCK	(Note 10)

CV_3_CAP	X	X	X	X	X	CMU84, QSU	X
HFCV_3_CAP	X	X	X	X	X	CMU84	X
HFCV_CORR_3_CAP	X	X				CMU84	X
JRAMP_3_CAP	X	X	X	X	X		
JTDDB_3_CAP	X	X	X	X	X		
MBLION_3_CAP	X	X	X	X	X	QSU	X
QCV_3_CAP	X	X	X	X	X	QSU	X
VRAMP_3_CAP	X	X	X	X	X		
VTDDB_3_CAP	X	X	X	X	X		
MBLION_SHS	X	X	X	X	X		
CPF_3_MOS	X	X	X	X	X	PGU	
CPV_3_MOS	X	X	X	X	X	PGU	
DELTAL_M2F	X	X	X	X	X		
DOP_MOS	X	X	X	X	X		
HCI_MOS	X			X	X		
HCI_3_MOS	X				X		
HCSTRESS_MOS	X	X			X		
STRSMEAS_2_MOS	X	X			X		
IDIBIG_MOS	X	X			X		
IDVD_MOS	X	X	X	X	X		
MAXSTRS_MOS	X	X	X	X	X		
S_MOS	X	X	X	X	X		
VPT_MOS	X	X	X	X	X		
VT_MOS	X	X	X	X	X		
CALC_ID_MOS	X						
CALC_VT_MOS	X						
LIFETIME_2_MOS	X		X	X	X		

HC_LINK_2_MOS					X		
PD_LINK_MOS					X		
RECORD_DATA[4,5,6]							
PLAY_DATA[4,5,6]							
GET_WLR_FLAG[4,6]							
PDQ_DB_ON[4,7]	(Note 8)				(Note 9)		
PDQ_DB_OFF[4,7]							

1. "Connect DUT" means that the algorithm will perform a DUT connection and measurement and is compatible with the Agilent 4062UX and Agilent 4070 series parametric testers
2. External equipment is identified by class, as follows:
   - TCHUCK - prober thermochuck - only the Temptronics 314B and command-compatible units supported at this time; OEM thermochucks for Electroglas, TSK, and TEL
   - CMU84 - precision LCR meter - only the Agilent 4284A is supported at this time
   - QSU - picoamp meter - only the Agilent 4140B is supported at this time
   - PGU - pulse generator - the Agilent 8110A and the Agilent 8160A are supported at this time
3. "Controls Prober" means that the algorithm requires the chuck to be raised and lowered, and may require that the DUT be bathed in light during certain portions of the test. PDQ-WLR can be configured for automatic prober control, or can be configured to pause during test execution with a prompt for user intervention at the prober.
4. These algorithms set the internal state of PDQ-WLR TIS or database operations; that is, they control the behavior of other algorithms.
5. The Record/Playback measurement feature is currently supported only on the Agilent 4062UX; no action is taken if these algorithms are used with the Agilent 4070 series.
6. Separate HP BASIC files do not exist for Record/Playback or reading algFlags using GET_WLR_FLAG. The actual subprograms are embedded in /opt/WLR/lib/WLRLibSupport.
7. Separate HP BASIC files do not exist for database control. The actual subprograms are embedded in /opt/WLR/lib/WLRdbf_mgmt.
8. PDQ_DB_ON performs implicit input error-checking. If the user specifies an illegal HP-UX path or file name, such as including illegal characters, the PDQ-WLR database management routines will substitute underscores, and attempt to use the replacement name. If the path and filename cannot be resolved, or the user attempts to create a new directory at the root level, the PDQ-WLR database management routines will revert to the system defaults in /var/opt/WLR/database/data/measured for data storage.
9. PDQ_DB_ON does not output data directly, but allows other algorithms to output data to the database. The default is for the database to be on unless explicitly turned off using PDQ_DB_OFF.
10. Thermochucks are a prober attachment.

Algorithm Methodologies

Methodologies for individual algorithms are covered in detail in Chapters 4, 5, and 6 of the PDQ-WLR User's Guide, and are not repeated here. Instead, this section deals with the overall approach by algorithm class. The type of failure mechanism found for any wafer depends upon the specific test structure, specific algorithm, and test constraints (expressed in the user inputs).

The EMR algorithms essentially revolve around a JEDEC-compliant Kelvin resistance (KELRES) measurement technique. A KELRES measurement is made for the DUT in a quiescent state. Following that, an iterative process of applying a stress and performing a high-current Kelvin resistance measurement occurs. Results are tracked in real-time; depending upon the specific algorithm and the user inputs, the real-time results are used to flag and classify failed DUTs. Failure classification is given in the same format for all EMR algorithms.

The MOS routines essentially revolve around TIS Set/Sweep measurements, where either a voltage or current is applied to the DUT, and a voltage or current is measured (not necessarily on the same pin as the stress application). An exception is the charge pumping algorithms, which use a pulse generator in addition to the normal compliment of Agilent parametric tester equipment. Currently supported pulse generators are the Agilent 8110A and the Agilent 8160A; these are programmed to provide trapezoidal and triangular waveforms as a signal source. Together with the computational class of algorithms, the hot carrier algorithms are used to make lifetime predictions about a wafer. By constraining the user inputs appropriately, many of these algorithms can serve as simple parametric tests.

The oxide algorithms have three subclasses; one may be used to drive a DUT to failure, one is used to perform various CV measurements, and one is used with a special self-heating test structure to detect mobile ion contamination.

The first subclass consists of JEDEC-compliant voltage and current ramp-to-failure algorithms and voltage and current time dependent dielectric breakdown (TDDB) algorithms. These operate using TIS Force/Measure command pairs for the ramps, and a constant Force with iterative

measurements for the TDDB codes. JEDEC-type return codes are consistently implemented for both ramp and TDDB codes; the returns cannot be identical due to the differences in measurement types.

The CV-type measurements rely upon an external LCR meter and an external picoamp meter. Currently the Agilent 4284A, for high frequency CV (HFCV), and the Agilent 4140B, for quasi-static CV (QCV) type measurements, are supported by PDQ-WLR. The HFCV measurements are accomplished with Force/Measure command pairs. These commands are not an integral part of the TIS, and must be bound in to the software system from the Agilent-supplied 4284 support library; the PDQ-WLR4284 instrument library provides this support by binding in a direct copy of the Agilent 4284 driver library. The QCV-type measurements are accomplished using Set/Sweep command pairs coded into the PDQ-WLR library distributed for Agilent 4140B support. These routines begin by performing a residual capacitance measurement which is accomplished by issuing prober commands to remove the DUT or by prompting the user to do so, depending upon the override flag. These routines also require a light source to provide electron-hole pairs when sweeping from inversion. At present, Agilent SPECS and HP BASIC users must provide the correct calls to accomplish this or rely on the appropriate override to prompt the user to cycle the light source manually. By constraining the user inputs appropriately, these algorithms can also serve as simple parametric tests.

The data management algorithms are unique in that their source location is obscured from the user. Rather than reside as separate files in the same directory as the WLR algorithms, these subprograms are stored within one of the support libraries. There are two reasons for this. First, we do not want to force HP BASIC users to have to manage separate files that may be used frequently; instead, it's more natural to think of these capabilities as library functions. Second, we do not want these files to be easily modified. We anticipate users freely exploring the HP BASIC algorithms to change measurement characteristics. We expect that library modifications by users will be rare; the activity has a higher risk factor because, in general, the libraries contain code shared by many algorithms. Without exact knowledge of what's being done, a library change to "fix" something is very liable to break several things. Users considering a library modification should proceed with that warning in mind.

Single-Point Maintenance

Philosophy and Implementation

PDQ-WLR was developed under the following design rules:

First design rule:

Only one distribution required to support any combination of user-interface or Agilent parametric tester. The possibilities supported are:

- New WLR users beginning with PDQ-WLR
- Existing users upgrading from SWORD to PDQ-WLR
- Agilent SPECS test shell
- Custom test software developed in HP BASIC
- The Agilent 4070 series parametric tester
- The Agilent 4062UX parametric tester

Second design rule:

No code shall be duplicated to meet the first design rule. This means that if a user is building custom test software using HP BASIC, modifies code, and then migrates to Agilent SPECS later, the changes may be retained, at the user's option. This is called "single-point maintenance". This rule implies, and IS MEANT TO IMPLY, that any changes made by the factory to a code (based on analysis, inspection, testing, design reviews, etc.) are immediately and automatically incorporated into all user-versions by a single change. (For example, a VRAMP change occurs in only one place; there is no danger of the "Agilent SPECS VRAMP" and the "HP BASIC VRAMP" falling out of phase with one another.)

Third design rule:

Any and all existing (SWORD-based) custom test software written in HP BASIC will need to run without modification, provided that the SWORD user did not modify any of the SWORD WLR algorithms' argument lists. With the exception of the libraries, which contain private methods, everything is preserved - even the site-specific scripts and build procedures which use the PDQ-WLR makewlrprog script.

This is accomplished with what we call a series of "**translation layers**" in software. The translation layers are different depending upon what user protection was targeted and what system is in use.

There are two types of translation layers:

- The Agilent SPECS Translation Layer
- The SWORD Translation Layer

It's helpful to first understand the algorithm naming conventions before going on to translation layers.

Algorithm Naming Conventions:

In order to differentiate the PDQ-WLR development from the last major release of SWORD (Version 2), PDQ-WLR introduced a generation number. This number is different than the Agilent release number. In September 1998, the PDQ-WLR generation number was 3 for the Agilent release of PDQ-WLR A.01.00. This is important, because the generation number contributes to the naming convention.

In SWORD, going from 1.0 to 2.0, algorithm changes were marked by putting an "X" in front of the name, so VRAMP_CAP became XVRAMP_CAP, ISO_EMR became XISO_EMR, etc. That naming convention would rapidly degenerate when letter-codes were exhausted. This was especially important

in light of the Third Design Rule protecting SWORD testplans. The final decision was to drop the character prefixes. Instead, for algorithms whose inputs/outputs changed, the adopted convention becomes:

<algorithm-root-name>_<generation-number>_<test-family>

So, VRAMP_3_CAP is the JEDEC-compliant VRAMP test from PDQ-WLR, generation 3, used for testing oxides.

For algorithms whose inputs/outputs remained unchanged, the original naming convention was retained:

<algorithm-root-name>_<test-family>

regardless of whether algorithm improvements were made, as was the case with all algorithms.

In this version of PDQ-WLR, the EMR codes are now in their third generation with the exception of the isothermal routine, now in its fourth generation. The gate oxide routines also in their third generation, with the exception of MBLION_SHS, still in its first generation. The MOSFET routines are primarily in their first generation, with the exception of the second generation routines for STRSMEAS, LIFETIME, and HC_LINK.

PDQ-WLR algorithms are subject to continued refinement. The generation number doesn't change until the input/output parameter list changes. Internal improvements are tracked via the embedded configuration management version identification for each algorithm.

Agilent SPECS Translation Layer

The Agilent SPECS Translation Layer translates algorithm names only. All user inputs and outputs are transported exactly between the PDQ-WLR root algorithms and the Agilent SPECS environment, with the exception of the no obsolete HCI_MOS, which requires an array dimensional-translation of the Hci_data for test plan compatibility.

The algorithm library specifications (test specs) all call routines with the general name format of

<PDQ-WLR_root_algorithm_name>_XL

where "_XL" stands for translation layer. Unfortunately, due to other system limitations, there are some exceptions to this. First, HP BASIC has a 15 character limit for all symbols; so HFCV_CORR_3_CAP_XL became HFCV_CORR3CAP_X. Second, HP BASIC generates an error if the line is too complex. Because of this, BEM_3_EMR_XL had to be shorted to BEM_3_EMR_X, and CV_3_CAP_XL and HFCV_3_CAP_XL had to have all inputs mapped to single characters.

The following table shows the mapping between the root algorithm names and the Agilent SPECS algorithm names. Note that this table is arranged alphabetically by class, whereas the Agilent SPECS Assist window presents the algorithms in their most commonly expected order-of-use. The algorithm explanations throughout the PDQ-WLR User's Guide also follows the order-of-use.

PDQ-WLR to Agilent SPECS Translation Names

PDQ-WLR Root Algorithm Name	Agilent SPECS Translation Layer Name (Appears in Agilent SPECS Assist window when selecting an algorithm)
BEM_3_EMR	BEM_3_EMR_X
CONST_I_3_EMR	CONST_I_3_EMR_X
ISO_4_EMR	ISO_4_EMR_XL
ISO_3_EMR	ISO_3_EMR_XL
KELRES_3_EMR	KELRES_3_EMR_XL
SWEAT_3_EMR	SWEAT_3_E_XL
TCR_3_EMR	TCR_3_EMR_XL
TVP_3_EMR	TVP_3_EMR_XL
CV_3_CAP	CV_3_CAP_XL
HFCV_3_CAP	HFCV_3_CAP_XL
HFCV_CORR_3_CAP	HFCV_CORR3CAP_X
JRAMP_3_CAP	JRAMP_3_CAP_XL
JTDDB_3_CAP	JTDDB_3_CAP_XL
MBLION_3_CAP	MBLION_3_CAP_XL
QCV_3_CAP	QCV_3_CAP_XL
VRAMP_3_CAP	VRAMP_3_CAP_XL
VTDDB_3_CAP	VTDDB_3_CAP_XL
MBLION_SHS	MBLION_SHS_XL
CPF_3_MOS	CPF_3_MOS_XL
CPV_3_MOS	CPV_3_MOS_XL
DELTAL_M2F	DELTAL_M2F_XL
DOP_MOS	DOP_MOS_XL
HCI_3_MOS	HCI_3_MOS_XL
STRSMEAS_2_MOS	STRSMS_2_MOS_XL
STRSMEAS_MOS	STRSMEAS_MOS_XL
HCI_MOS	HCI_MOS_XL

HCSTRESS_MOS	HCSTRESS_MOS_XL
IDIBIG_MOS	IDIBIG_MOS_XL
IDVD_MOS	IDVD_MOS_XL
MAXSTRS_MOS	MAXSTRS_MOS_XL
S_MOS	S_MOS_XL
VPT_MOS	VPT_MOS_XL
VT_MOS	VT_MOS_XL
CALC_ID_MOS	CALC_ID_MOS_XL
CALC_VT_MOS	CALC_VT_MOS_XL
LIFETIME_2_MOS	LIFETM_2__MOS_XL
LIFETIME_MOS	LIFETIME_MOS_XL
HC_LINK_2_MOS	HC_LINK_2_MOS_X
HC_LINK_MOS	HC_LINK_MOS_XL
PD_LINK_MOS	PD_LINK_MOS_XL
RECORD_DATA	RECORD_DATA_XL
PLAY_DATA	PLAY_DATA_XL
GET_WLR_FLAG	GET_WLR_FLAG_XL
PDQ_DB_ON	PDQ_DB_ON_XL
PDQ_DB_OFF	PDQ_DB_OFF_XL
SET_TCHUCK	SET_TCHUCK_XL

SWORD Translation Layer

The SWORD Translation Layer translates calls to obsolete SWORD algorithms into supported PDQ-WLR algorithms. In some cases, the obsolete SWORD routines did not accept inputs required by PDQ-WLR. For these cases, PDQ-WLR default input values are inserted in the translation process. In other cases, the PDQ-WLR algorithms produce outputs which have no counterpart in the obsolete SWORD algorithms. In this event, the new outputs are discarded. For this reason, users should migrate away from SWORD Translation Layer use as soon as possible, to gain the complete advantages of PDQ-WLR use.

For all algorithms, the resulting outputs will be different in many places due to computation improvements and defect corrections.

Normally, HP BASIC users would not need a translation layer; the root algorithms are loadsub'd and called directly. During PDQ-WLR development, it was determined that many SWORD algorithms had poor input/output argument lists; some lacked fundamental user inputs or outputs required for JEDEC compliance, others, such as the EMR family, did similar things with dissimilar - and therefore hard to analyze - argument lists. For these, a new root algorithm would be created, and the old algorithm became a "shell" around the new algorithm (the fundamental definition of a translation layer). The following is an example, using XVRAMP_CAP. XVRAMP_CAP calls VRAMP_3_CAP. To do so, it accepts the user inputs to XVRAMP, maps them to VRAMP_3 inputs, adds default values to the new VRAMP_3 inputs (such as the flag to test in inversion or test in accumulation) which will cause VRAMP_3 to exhibit the same functionality as XVRAMP (i.e., test in accumulation). Next, it tests to see if VRAMP_3_CAP is in memory. For HP BASIC users who built their software with a older makewlrprog based script which only called XVRAMP_CAP, the VRAMP_3_CAP routine would be missing. In that case, the translation layer loadsubs VRAMP_3_CAP from /opt/WLR/BASIC; by design, no attempt is made to search elsewhere for the file. With the inputs ready and the root algorithm in memory, the root algorithm is then called. Upon return, the VRAMP_3_CAP outputs are mapped into the XVRAMP_CAP outputs. Unused VRAMP_3_CAP outputs are discarded. Error return codes for PDQ-WLR are now standardized; in SWORD they were not. So, where applicable, error return codes are translated from the PDQ-WLR format into the format for the algorithm in question (in this case, XVRAMP).

SWORD Translation Layer Exceptions:

- The EMR set includes ISO_EMR, XISO_EMR, ISO_3_EMR and ISO_4_EMR codes. Of these, XISO and ISO_3 are translation layers. ISO_EMR was so different in nature that it could not be accommodated using the translation layer scheme. ISO_EMR is completely obsolete. It has been maintained only to fix obvious defects found in isothermal testing. It will be completely dropped in the next major release of PDQ-WLR. ISO_EMR exists to protect existing test plans only. SWORD 2.01 users were advised to migrate to XISO, which is now supported with a translation to ISO_3_EMR. XISO called ISO_3 as of PDQ-WLR 1.0 and now ISO_3 calls ISO_4. XISO is therefore obsolete by two generations and, like ISO and ISO_3, should be avoided in favor of using ISO_4 directly.
- VRAMP_CAP was eliminated. In light of the latest JEDEC requirements, its limited argument list, and the failure detection methods it employed, it was deemed not maintainable. SWORD 2.01 users were advised to migrate to XVRAMP_CAP, which is now supported with a translation to VRAMP_3_CAP.

The following table shows the mapping between the root algorithm names and the obsolete SWORD algorithm names. Note that this table is arranged alphabetically by class.

PDQ-WLR to SWORD Translation Names

PDQ-WLR Root Algorithm Name	SWORD Translation Layer Name (Root algorithms not appearing in this list do not require a translation.)
BEM_3_EMR	BEM_EMR
CONST_I_3_EMR	CONST_I_EMR
ISO_3_EMR	XISO_EMR
ISO_4_EMR	ISO_3_EMR
KELRES_3_EMR	KELRES_EMR
SWEAT_3_EMR	SWEAT_EMR
TCR_3_EMR	TCR_EMR
TCR_3_EMR	XTCR_EMR
TVP_3_EMR	TVP_EMR
CV_3_CAP	CV_CAP
HFCV_3_CAP	HFCV_CAP
HFCV_CORR_3_CAP	HFCV_CORR_CAP
JRAMP_3_CAP	JRAMP_CAP
JTDDB_3_CAP	TDDBI_CAP
MBLION_3_CAP	MBLION_CAP
QCV_3_CAP	QCV_CAP
VRAMP_3_CAP	XVRAMP_CAP
VTDDB_3_CAP	TDDBN_CAP
CPF_3_MOS	CPF_MOS
CPV_3_MOS	CPV_MOS
LIFETIME_2_MOS	LIFETIME_MOS

Directory Structure

PDQ-WLR always installs in the /opt hierarchy, under /opt/WLR. All files and directories are owned by the root user; group access is via the group users. The directory structure for the installed PDQ-WLR follows:

/opt/WLR/BASIC: *(Directory permission is rwxrwxrwx.)*
The core PDQ-WLR algorithms, in RMB LIF files.
The permission for all files is rw-rw-rw-.

Files:	Description:
BEM_3_EMR	Breakdown Energy of Metals
BEM_EMR	-- Thin layer to BEM_3_EMR --
CALC_ID_MOS	Calculate Id Ib Ig
CALC_VT_MOS	Calculate Vth
CONST_I_3_EMR	Constant Current
CONST_I_EMR	-- Thin layer to CONST_I_3_EMR --
CPF_3_MOS	Charge Pumping by Frequency
CPF_MOS	-- Thin layer to CPF_3_MOS --
CPV_3_MOS	Charge Pumping by Voltage
CPV_MOS	-- Thin layer to CPV_3_MOS --
CV_3_CAP	Calculate Dit at Vfb,Vmg,QSmin
CV_CAP	-- Thin layer to CV_3_CAP --
DELTAL_M2F	Delta Length, Drawn vs. Actual
DOP_MOS	Substrate Doping Concentration
HCI_3_MOS	Hot Carrier Injection
HCI_MOS	Hot Carrier Injection
HCSTRESS_MOS	Hot Carrier Stress
HC_LINK_2_MOS	Generic table used to link data stores
HC_LINK_MOS	Obsolete version of HC_LINK_2_MOS
HFCV_3_CAP	High Frequency Capacitance vs. Voltage
HFCV_CAP	-- Thin layer to HFCV_3_CAP --
HFCV_CORR_3_CAP	Test Fixture Configuration
HFCV_CORR_CAP	-- Thin layer to HFCV_CORR_3_CAP --
IDIBIG_MOS	MOS FET Currents
IDVD_MOS	MOSFET Drain I and V vs. Gate V
ISO_4_EMR	Isothermal
ISO_3_EMR	-- Thin layer to ISO_4_EMR --
ISO_EMR	Isothermal (Obsolete)
JRAMP_3_CAP	Current Ramp
JRAMP_CAP	-- Thin layer to JRAMP_3_CAP --
JTDDB_3_CAP	Constant Current TDDB

```
    KELRES_3_EMR            Kelvin Resistance Test
    KELRES_EMR              -- Thin layer to KELRES_3_EMR --
    LIFETIME_2_MOS          Lifetime plot
    LIFETIME_MOS            -- Thin layer to LIFETIME_2_MOS --
    MAXSTRS_MOS             Maximum Stress
    MBLION_3_CAP            Mobile Ions
    MBLION_CAP              -- Thin layer to MBLION_3_CAP --
    MBLION_SHS              Mobile Ions, self-heated structure
    PD_LINK_MOS             Plasma damage database store
    QCV_3_CAP               Quasi-static Capacitance vs. Voltage
    QCV_CAP                 -- Thin layer to QCV_3_CAP --
    STRSMEAS_2_MOS          Hot Carrier Stress
    STRSMEAS_MOS            -- Thin layer to STRSMEAS_2_MOS --
    SWEAT_3_EMR             SWEAT EM Test
    SWEAT_EMR               -- Thin layer to SWEAT_3_EMR --
    S_MOS                   Subthreshold slope
    TCR_3_EMR               Temperature Coeff Resistance
    TCR_EMR                 -- Thin layer to TCR_3_EMR --
    TDDBI_CAP               -- Thin layer to JTDDB_3_CAP --
    TDDBN_CAP               -- Thin layer to VTDDB_3_CAP --
    TVP_3_EMR               Temperature vs Power
    TVP_EMR                 -- Thin layer to TVP_3_EMR --
    VPT_MOS                 Voltage PunchThrough
    VRAMP_3_CAP             Voltage Ramp
    VTDDB_3_CAP             Constant Voltage TDDB
    VT_MOS                  MOSFET Vt, Gm, and Id
    XISO_EMR                -- Thin layer to ISO_3_EMR --
    XTCR_EMR                -- Thin layer to TCR_3_EMR --
    XVRAMP_CAP              -- Thin layer to VRAMP_3_CAP --
```

/opt/WLR/bin: *(Directory permission is rwxr-xr-x.)*

Presently, only the build program for BASIC users resides here, and the HCI_3_MOS Builder.

The file permissions are r-xr-xr-x.

Files:

makewlrprog

xhci

/opt/WLR/etc: *(Directory permission is rwxrwxr-x.)*

Configuration files used by the build scripts and for run-time overrides.

The file permissions are rw-rw-rw-.

```
    Files:
    WLR.config   algFlags
```

/opt/WLR/lib: *(Directory permission is rwxrwxrwx.)*
Instrument and calculational libraries required by PDQ-WLR algorithms.
The file permissions are rw-rw-rw-.

```
    Files:
    HCI_MOS_Lib                  No_HCI_MOS_Lib
    WLRHP4140B                   WLRHP4284A
    WLRHP5316B                   WLRHP8110A
    WLRHP8160A                   WLRLibSupport
    WLRLibcvanal                 WLRLibhci
    WLRLibmath                   WLRLibmisc
    WLRNoFreqCounter             WLRNoLCRmeter
    WLRNoPAmeter                 WLRNoPulseGen
    WLRNoThermochuck             WLRTEMP314B
    WLR_BASIC_Prober_template    WLR_BASIC_db_template
    WLR_SPECS_Prober             WLR_SPECS_db
    WLRdbf_mgmt                  WLRgraph_csv
    ICMSCOMPAT
    WLRTELP8_CHUCK               WLRSPECS_CHUCK
    WLETSK90A_CHUCK              WLREGCHUCK
```

/opt/WLR/local: *(Directory permission is rwxrwxrwx.)*
Created by the installation script as a convenience for HP BASIC and PDQ-WLR HCI_MOS users. A general-purpose, WLR-related user storage area, and also the installed location for the ALL_HCI subprogram for use with HCI_MOS.
The file permission is rw-rw-rw-.

```
    File:
    ALL_HCI                  User-specified HCI test controller
```

/opt/WLR/newconfig: *(Directory permission is rwxr-xr-x.)*
An on-line backup area for support purposes; kept read-only. This area is primarily intended to free users from having to re-install the product, or refer to backup tapes, in the event that a critical file becomes corrupted. Note that PDQ-WLR source code is NOT backed up here. Also note that because this is a read-only area, the REVISION file, used by the Agilent SPECS build script, is kept here. The current

product README file is also kept here.
The file permissions are r--r--r--, subdirectory permissions are rwx-r-xr-x.

Files:

README	Special_Notes	Xgraph	revision

Subdirectories:

.support	SPECS	samples

/opt/WLR/newconfig/.support: *(Directory permission is rwxr-xr-x.)*
Test algorithms for support personnel, not part of the PDQ-WLR suite.
The file permissions are r--r--r--, the subdirectory permission permission is rwxr-xr-x.

Files:

CP3.MOS	CP3_MOS.A	CPTRTF.MOS	CPTRTF_MOS.A
HCI_M5F.A	LINWD.XBR	LINWD_XBR.A	M3FET
M4FET	M5FET	README.support	RECOMB.CAP
RECOMB_CAP.A	SETTEMP	SETTEMP.A	XBRIDGE

Subdirectory:
Zdocumentation

/opt/WLR/newconfig/.support/Zdocumentation:
(Directory permission is rwxr-xr-x.)
Compressed PostScript documentation for algorithms in the .support directory, above.
The file permissions are r--r--r--.

Files:

CP3.ps.Z	CPTRTF.ps.Z

/opt/WLR/newconfig/SPECS: *(Directory permission is rwxr-xr-x.)*
Backup copies of PDQ-WLR/Agilent SPECS libraries and specification files.
The subdirectory permissions are r--r--r--.

/opt/WLR/newconfig/SPECS/alg: *(Directory permission is rwxr-xr-x.)*
The PDQ-WLR .lib and .bas files, copied to the /opt/SPECS/usr/alg/measure

area by the installation script. Note that the .bas file contains only the algorithm thin-layers, not actual algorithms. The file permissions are r--r--r--.

Files:
PDQ_WLR3_00.bas PDQ_WLR3_00.lib *(Name reflects major release.)*

/opt/WLR/newconfig/SPECS/alg_library:
(Directory permission is rwxr-xr-x.)
Individual thin-layer algorithm specification files. Not normally used, but provided as a convenience for advanced users capable of building their own Agilent SPECS libraries; these files provide PDQ-WLR component parts for those cases. The file permissions are r--r--r--.

Files:

BEM_3_EMR_X.alg	CALC_ID_MOS_XL.alg	CALC_VT_MOS_XL.alg
CONST_I_3_EMR_X.alg	CPF_3_MOS_XL.alg	CPV_3_MOS_XL.alg
CV_3_CAP_XL.alg	DELTAL_M2F_XL.alg	DOP_MOS_XL.alg
GET_WLR_FLAG_XL.alg	HCI_MOS_XL.alg	HCSTRESS_MOS_XL.alg
HC_LINK_MOS_XL.alg	HFCV_3_CAP_XL.alg	HFCV_CORR3CAP_X.alg
IDIBIG_MOS_XL.alg	IDVD_MOS_XL.alg	ISO_3_EMR_XL.alg
JRAMP_3_CAP_XL.alg	JTDDB_3_CAP_XL.alg	KELRES_3_EMR_XL.alg
LIFETIME_MOS_XL.alg	MAXSTRS_MOS_XL.alg	MBLION_3_CAP_XL.alg
PDQ_DB_OFF_XL.alg	PDQ_DB_ON_XL.alg	PLAY_DATA_XL.alg
QCV_3_CAP_XL.alg	RECORD_DATA_XL.alg	SWEAT_3_E_XL.alg
S_MOS_XL.alg	TCR_3_EMR_XL.alg	TVP_3_EMR_XL.alg
VPT_MOS_XL.alg	VRAMP_3_CAP_XL.alg	VTDDB_3_CAP_XL.alg
STRAMEAS_MOS_XL.alg	MBLION_SHS_XL.alg	PD_LINK_MOS.alg
HCI_3_MOS_XL.alg	ISO_4_EMR_XL.alg	LIFETM_2_MOS_XL.alg
STRSMS_2_MOS_XL.alg	HC_LINK_2_MOS_X.alg	VT_MOS_XL.alg

/opt/WLR/newconfig/SPECS/bin: *(Directory permission is rwxr-xr-x.)*
Backup of the PDQ-WLR/Agilent SPECS build script, makePDQ-WLRLib, which is copied to the SPECS/usr/bin area by the installation script. A copy of Agilent SPECS 2.2 BASIC.make, provided by Agilent, is kept here; it is executed from this area when Agilent SPECS 2.1 users run /SPECS/usr/bin/makePDQ-WLRLib.
The file permissions are r-xr-xr-x.

Files:

BASIC.make	makePDQ-WLRLib	ExtendPDQLib

ExtendPDQLib support files are also here.

/opt/WLR/newconfig/SPECS/fwk: *(Directory permission is rwxr-xr-x.)*
A sample framework; working copy at SPECS/usr/fwk.
The file permission is r--r--r--.

File:
PDQWLR.fwk wlr_fwk_tpt_final.tar

/opt/WLR/newconfig/etc: *(Directory permission is rwxr-xr-x.)*
Backups of the working files in /opt/WLR/etc; algFlags and WLR.config.
The file permissions are r--r--r--.

Files:
WLR.config algFlags

/opt/WLR/newconfig/samples: *(Directory permission is rwxr-xr-x.)*
Sample programs and a README for hot carrier tests.
The file permissions are r--r--r--.

Files:

ALL_HCI.A	README	VT1_HCI.A	all_hci.input
pretest_n_6	pretest_p_5	qual_ex_n_33	qual_ex_n_25
qual_ex_n_50			

Libraries

This section describes the libraries by type and also describes the purpose of each subprogram within each library.

Each library begins with a subprogram called ***Rcs_id_<number>***. These subprograms do not contain any executable statements and exist only to provide a container for revision control system (RCS) identification data, which appears in the form of HP BASIC comments. The number suffix on the name is unimportant and is used only to give each file's RCS identification container a unique name. The numbers themselves are auto-generated during development, and should never be used to identify a file or its version; instead use the information found within the container. The revision control system stamps each container with the individual version number of the file, the last revision date of the file, and the file identifier within the ST revision control system. These containers exist so that the software engineering management data is kept safely apart from the executable code. The identifiers are sufficient to allow tracking of any individual file by Agilent personnel.

Libraries are accessible by one of three methods, depending upon the type of user.

- For Agilent SPECS users, the libraries become part of the Agilent SPECS BASIC Interpreter. This is done by executing the makePDQ-WLRLib PDQ-WLR script, which in turn executes the BASIC.make Agilent SPECS script.
- For HP BASIC users, the makewlrprog PDQ-WLR script assembles the user's main program, PDQ-WLR algorithms, and PDQ-WLR libraries into a single HP BASIC file. Even simple user programs will have relatively large file sizes because of this.

For the purposes of this discussion, we'll refer to any of these access methods as part of an execution profile.

There are four types of libraries in PDQ-WLR:

- Instrument Libraries
- Computational Support Libraries
- Data Output Support Libraries
- Agilent IC-MS Compatibility Library

PDQ-WLR Algorithms vs. Library Use

The following table shows which libraries are required to support each PDQ-WLR algorithm.

Algorithm	TIS WLRLibmisc	WLRLibSupport	WLRLibmath	WLRLibcvanal	WLRLibhci	WLRgraph_csv	WLRdbf_mgmt WLR_SPECS_db WLR_BASIC_db	External Equipment Library Class Name	Prober Interface Library	ICMSCOMPAT
BEM_3_EMR	X	X	X			X	X			X
CONST_I_3_EMR	X	X				X	X			X
ISO_4_EMR	X	X	X			X	X			X
KELRES_3_EMR	X	X					X			X
SWEAT_3_EMR	X	X				X	X			X
SET_TCHUCK	X	X						TCHUCK		
TCR_3_EMR	X	X	X			X	X	TCHUCK		X
TVP_3_EMR	X	X	X			X	X			X
CV_3_CAP		X		X		X	X	CMU84, QSU	X	X
HFCV_3_CAP	X	X		X		X	X	CMU84	X	X

HFCV_CORR_3_CAP	X	X						CMU84	X	X
JRAMP_3_CAP	X	X				X	X			X
JTDDB_3_CAP	X	X				X	X			X
MBLION_3_CAP	X	X				X	X	QSU	X	X
QCV_3_CAP	X	X				X	X	QSU	X	X
VRAMP_3_CAP	X	X	X			X	X			X
VTDDB_3_CAP	X	X				X	X			X
MBLION_SHS	X	X				X	X			X
CPF_3_MOS	X	X	X			X	X	PGU		X
CPV_3_MOS	X	X	X			X	X	PGU		X
DELTAL_M2F	X	X	X			X	X			X
DOP_MOS	X	X	X			X	X			X
STRSMEAS_MOS	X	X					X			X
HCI_MOS		X				X	X			X
HCSTRESS_MOS	X	X					X			X
IDIBIG_MOS	X	X					X			X
IDVD_MOS	X	X	X			X	X			X
MAXSTRS_MOS	X	X				X	X			X
S_MOS	X	X	X			X	X			X
VPT_MOS	X	X				X	X			X
VT_MOS	X	X	X			X	X			X
CALC_ID_MOS		X								X
CALC_VT_MOS		X								X
LIFETIME_MOS		X	X			X	X			X
HC_LINK_MOS							X			
PD_LINK_MOS							X			
ALL_HCI[1]					X					
HCI_3_MOS					X					

1. The ALL_HCI sample hot carrier test algorithm, used with HCI_MOS, is a control manager for hot carrier measurement algorithms; additional library requirements are set by those algorithms.

Instrument Libraries

There are two types of Instrument Libraries: prober interfaces and external equipment drivers.

Prober Interfaces for CV Users

Normally, prober control is provided by the test shell (Agilent SPECS) or by user-defined routines (HP BASIC users). The CV algorithms require prober control in order to temporarily remove the DUT or to provide light injection. In order to accomplish this, PDQ-WLR algorithms call central routines contained in the WLRLibSupport library. Depending upon the overrides set by the user in the /opt/WLR/etc/algFlags file, either a prober interface library is called, or the program pauses for user-intervention. If the algFlags "Auto Prober" entry is on, the WLRLibSupport library calls routines in the user's PDQ-WLR prober interface library, which must be specified by the user in /opt/WLR/etc/WLR.config. PDQ-WLR supplies one prober interface library and one template for building a prober interface library in /opt/WLR/lib: WLR_SPECS_Prober and WLR_BASIC_Prober_template. The WLR.config entries for these are:

- BLIB /opt/WLR/lib/WLR_SPECS_Prober
- BLIB /opt/WLR/lib/WLR_BASIC_Prober

HP BASIC users will be required to create a WLR_BASIC_Prober library from the PDQ-WLR template. The prober template includes calls to Pup, Pdown, Plight_on, and Plight_off. Pup and Pdown are already supported for many probers by the software provided with the Agilent parametric testers. If the prober to be driven is supported by an existing Agilent or user library, it must be specified in the WLR.config file. The following WLR.config entries show how to do this for an Electroglas prober on both supported testers:

- Agilent 4070 Series: BLIB /opt/hp4070/lib/EGX
- Agilent 4062UX: BLIB /usr/pcs/lib/EGX

If the algFlags "Prober Light" entry is on, then the user must provide the appropriate code in the PDQ-WLR prober interface library for automatic light control, under “Plight_on_xl” and “Plight_off_xl”.

The default WLR.config and algFlags specifications are to auto-probe using the WLR_SPECS_Prober library, with no automatically-controlled prober light.

The prober interface libraries are:

- WLR_SPECS_Prober
- WLR_BASIC_Prober_template

Prober Interface Libraries

```
Routines in WLR_SPECS_Prober:
  SUBPROGRAMS:
    SUB Rcs_id_28308                                    !File id
    SUB Pup_xl
                Raises the prober chuck.
    SUB Pdown_xl
                Lowers the prober chuck.
    SUB Plight_on_xl
                Turns on the prober light - must be supplied by the
                user at this time.
    SUB Plight_off_xl
                Turns off the prober light - must be supplied by the
                user at this time.


Routines in WLR_BASIC_Prober_template:
  SUBPROGRAMS:
    SUB Rcs_id_28306                                    !File id
    SUB Pup_xl
    SUB Pdown_xl
    SUB Plight_on_xl
    SUB Plight_off_xl
                These four subprograms are templates for the HP BASIC
                user to use to create a site-specific PDQ-WLR prober
                interface.
```

External Equipment Drivers

PDQ-WLR algorithms may rely upon equipment not normally included with Agilent's parametric test systems. PDQ-WLR algorithms are NOT permitted to issue direct instrument commands; instead, algorithms must refer to an instrument library for external equipment management.

The choice for which instrument libraries to include in a user's execution profile is determined by the /opt/WLR/etc/WLR.config file. As noted in Chapter 1, the WLR.config file must be modified with a text editor to specify available hardware. If an instrument is defined as being part of the site configuration, then the /opt/WLR/lib area is scanned for the instrument library, using the naming convention of: WLR<*instrument-name*>. By following this naming convention, users are free to create and add new instrument libraries into their execution profiles.

For each instrument class known to PDQ-WLR, there exists a library which denotes that the instrument in question is not part of the site configuration, as defined in the WLR.config file. The naming convention for these libraries is: WLRNo<*instrument-description*>. For each type of instrument ***not*** specified in the WLR.config file, a "WLR No" library is inserted. ***This is an important user feature because it will prevent an incorrectly configured system from locking up during testing***. If a user attempts to use an algorithm requiring an external instrument not properly configured the appropriate "WLR No" library will intercept the attempt, identify the problem by instrument type and PDQ-WLR algorithm, refer the user to the User's Guide and to the WLR.config file, and will stop the test. The entries in each "WLR No" instrument library listing, below, are not discussed because all such entries flow directly to the "Fatal_no_lib" subprogram in the WLRLibSupport computational support library.

The external equipment driver libraries, by class, are:

Class CMU84:

- WLRHP4284A
- WLRNoLCRmeter

Class QSU:

- WLRHP4140B
- WLRNoPAmeter

Class PGU:

- WLRHP8110A
- WLRHP8160A
- WLRNoPulseGen

Class TCHUCK:

- WLRTEMP314B
- WLRNoThermochuck
- WLRTELP8_CHUCK
- WLRTSK90A_CHUCK
- WLREGCHUCK
- WLRSPECS_CHUCK

Class FREQ:

- WLRHP5316B
- WLRNoFreqCounter

LCR Meter Libraries

PDQ-WLR Class CMU84

```
Routines in WLRHP4284A:
  SUBPROGRAMS:
    SUB Rcs_id_28291                                    !File id
    SUB Cv_cmu84_idn_
              Instrument identifier - returns "AT4284A".
    SUB Cv_cmu84_init
              Initializes the instrument.
    SUB Get_opt_at4284
              Queries the instrument for the installed options.
    SUB Set_correction
              Sets the cable length, open, and short correction
              parameters.
    SUB Dvr4284
              This routine, and the following five, are copies of
              Agilent-distributed DRVR4284 library, added here for
              user convenience in building algorithms.  The routines
              provide a translation between Agilent 4284A calls and
              TIS calls.  Refer to the BASIC Programming Guide for
              your parametric tester for details.
    SUB Init_cmu84
    SUB Frequency84
    SUB Force_v84
    SUB Disable_port84
    SUB Port_status84

Routines in WLRNoLCRmeter:
  SUBPROGRAMS:
    SUB Rcs_id_28301                                    !File id
    SUB Cv_cmu84_idn_
    SUB Cv_cmu84_init
    SUB Get_opt_at4284
    SUB Set_correction
    SUB Dvr4284
    SUB Init_cmu84
    SUB Frequency84
    SUB Force_v84
    SUB Disable_port84
   SUB Port_status84
```

Picoamp Meter Libraries

PDQ-WLR Class QSU

```
Routines in WLRHP4140B:
  COMMONS:
    COM /Hp4140_/
                Contains instrument state data and correction arrays.
  SUBPROGRAMS:
    SUB Rcs_id_28290                                    !File id
    SUB Qsu_idn
                Instrument identifier - returns  "AT4140B".
    SUB Init_qsu
                Clears the instrument and its GPIB interface.
    SUB Set_qsu
                Sets the signal-generation parameters to prepare
                for a sweep measurement.
    SUB Sweep_qsu
                Performs the sweep measurement and applies the
                measured correction factors to the sweep data.
    SUB Zero_qsu
                Performs a sweep under quiescent conditions to
                build up a correction factors array.
    SUB Swap_zero_qsu
                As a programming convenience, reverses the order
                of the correction factors array to support tests
                which require corrected sweeps in both directions.

Routines in WLRNoPAmeter:
  SUBPROGRAMS:
    SUB Rcs_id_28302                                    !File id
    SUB Qsu_idn
    SUB Init_qsu
    SUB Set_qsu
    SUB Sweep_qsu
    SUB Zero_qsu
```

Pulse Generator Libraries

PDQ-WLR Class PGU

```
Routines in WLRHP8110A:
  SUBPROGRAMS:
    SUB Rcs_id_28293                                    !File id
    SUB Pg_idn
              Instrument identifier - returns "AT8110A".
    SUB Pg_init
              Initializes the instrument.
    SUB Pg_triangle
              Sets the triangular waveform characterics.
    SUB Pg_set_v
              Sets the peak-to-peak signal voltage.
    SUB Pg_disable
              Disables the instrument's signal output.
    SUB Pg_enable
              Enables the instruments's signal output.
    SUB Pg_set_z
              Sets the output impedance of the instrument.
    SUB Pg_set_f
              Sets the waveform frequency, rise, and fall times.
  FUNCTIONS:
    DEF FNPg_scale$
              Used only internally by this library.  Scales values
              in engineering units to the correct command suffix.
```

```
Routines in WLRHP8160A:
              These subprograms provide the same services as those
              listed for the WLRHP8110A library, with the differences
              noted below.
  SUBPROGRAMS:
    SUB Rcs_id_28294                                  !File id
    SUB Pg_idn
              Returns "AT8160A".
    SUB Pg_init
    SUB Pg_triangle
    SUB Pg_set_v
    SUB Pg_disable
    SUB Pg_enable
    SUB Pg_set_z
    SUB Pg_set_f
  FUNCTIONS:
    DEF FNChan8160$
              Used only internally by this library.  Converts
              numerical channel id to string.

Routines in WLRNoPulseGen:
  SUBPROGRAMS:
    SUB Rcs_id_28303                                  !File id
    SUB Pg_idn
    SUB Pg_init
    SUB Pg_triangle
    SUB Pg_set_v
    SUB Pg_disable
    SUB Pg_enable
    SUB Pg_set_z
    SUB Pg_set_f
```

Thermochuck Control Libraries

PDQ-WLR Class TCHUCK

```
Routines in WLRTEMP314B:
  SUBPROGRAMS:
    SUB Rcs_id_28305                                  !File id
    SUB Tchuck_idn
                Instrument identifier - returns "TEMP314B".
    SUB Tchuck_set
                Sets and then confirms the thermochuck temperature.

The other thermochuck libraries have the same internal structure, but
return different identifiers.

Routines in WLRNoThermochuck:
  SUBPROGRAMS:
    SUB Rcs_id_28304                                  !File id
    SUB Tchuck_idn
    SUB Tchuck_set
```

Frequency Counter Libraries

PDQ-WLR Class FREQ

```
Routines in WLRHP5316B:
  SUBPROGRAMS:
    SUB Rcs_id_28292                                  !File id
    SUB Freq_idn
                Instrument identifier - returns "AT5316B".
    SUB Freq_init
                Initializes the instrument.
    SUB Freq_set_trig
                Sets the trigger threshold level.
    SUB Freq_read
                Returns the frequency counter value.

Routines in WLRNoFreqCounter:
  SUBPROGRAMS:
    SUB Rcs_id_28300                                  !File id
    SUB Freq_idn
    SUB Freq_init
    SUB Freq_set_trig
    SUB Freq_read
```

Computational Support Libraries

Computational support libraries are used by PDQ-WLR algorithms whenever repeated tasks, or standardized methods of operation, are required.

The computational support libraries are:

- WLRLibSupport
- WLRLibmisc
- WLRLibhci
- WLRLibmath
- WLRcvanal

WLRLibSupport

```
Routines in WLRLibSupport:
  COMMONS:
    COM /Cv_calcs/
    COM /Cv_data/
                Intermediate state and data results from CV measurements.
    COM /Wlrx_priv_srvr/
                State to indicate if the server flag data structures
                have been initialized with algFlags file values.
  SUBPROGRAMS:
    SUB Rcs_id_28295                                    !File id
    SUB Shfcv_cap
                Shared high frequency CV measurement primitives.
    SUB Sqscv_cap
                Shared quasi-static CV measurement primitives.
    SUB Disconnect_all
                Disconnects tester from DUT, and returns test hardware
                to the defined initial state.
    SUB Hci_savedata
                Primitive to store hot carrier measurement data in HCI_MOS
                array.
    SUB Hci_savestrs
                Primitive to store hot carrier stress data in HCI_MOS
                array.
    SUB Hci_getparm
                Returns requested parameter data from HCI_MOS measurement
                data array.
    SUB Wlrx_maxtimeout
                A translation layer for the ICMSCOMPAT maximum-time-before-
                test-timeout function.
    SUB Get_wlr_flag
                Rereads the algFlags file and refills the server flag
                array.
    SUB Fatal_no_lib
                A terminal point for all "WLR No Instrument" libraries.
    SUB St_plight_off
                A control-management layer between PDQ-WLR algorithms
                and the user prober interface to turn the prober light
                off.
    SUB St_plight_on
                A control-management layer between PDQ-WLR algorithms
                and the user prober interface to turn the prober light
                on.
    SUB St_pdown
                A control-management layer between PDQ-WLR algorithms
```

```
              and the user prober interface to lower the chuck.
    SUB St_pup
              A control-management layer between PDQ-WLR algorithms
              and the user prober interface to raise the chuck.
    SUB Xdialog_mesg
              Provides a test interruption calling for operator
              intervention.  The intervention message is passed in to
              the routine.
    SUB Record_data
              Makes a TIS call to store measurement data to the user-
              specified file.
    SUB Play_data
              Makes a TIS call to playback the user-specified measurement
              recording file, when the system is in the Offline Mode.
    SUB Setup_pdqwlr
              A single control point for PDQ-WLR algorithms to call to
              set the test and data system to required known states.
              Presently, only Set_opt_level is called.
    SUB Set_opt_level
              For Agilent 4070 series users, sets the tester optimization
              level for highest accuracy and external equipment
              synchronization.
    SUB Tester_control
              For PDQ-WLR algorithms which detect unconstrained test
              conditions, this routine presents a window with user
              instructions, performs an orderly hardware cleanup and
              DUT disconnect, and calls Test_control to terminate
              testing.
    SUB Compactify_str
              Removes leading, trailing, and internal spaces and tabs
              from a string.
    SUB Fix_illegal
              For path and filename parsing use; replaces the characters
              specified on input with an underscore in the provided
              path or filename string.
  FUNCTIONS:
    DEF FNWlrx_srvrflag
              A translation layer for returning the server flag value.
    DEF FNWlr_prober
              Returns a flag to indicate if an autoprober is in use.
    DEF FNSt_port
              A translation layer to make TIS FNPort calls.
```

WLRLibmisc

```
Routines in WLRLibmisc:
  SUBPROGRAMS:
    SUB Rcs_id_28299                                    !File id
    SUB Find_i_comp_
                Returns the minimum of a user input, or the compliance
                limit, for a specified voltage range.
    SUB Powcomp_i_
                Returns the current compliance for a given voltage for
                an SMU which PDQ-WLR algorithms assume is a high-power
                SMU; the actual configuration is defined in WLR.config.
    SUB Powcomp_i_atsmu
                Returns the high-power SMU current compliance for a
                given voltage.
    SUB Powcomp_i_mpsmu
                Returns the medium-power SMU current compliance for a
                given voltage.
    SUB Powcomp_v_
                Returns the voltage compliance for a given current for
                an SMU which PDQ-WLR algorithms assume is a high-power
                SMU; the actual configuration is defined in WLR.config.
    SUB Powcomp_v_atsmu
                Returns the high-power SMU voltage compliance for a
                given current.
    SUB Powcomp_v_mpsmu
                Returns the medium-power SMU voltage compliance for a
                given current.
    SUB Stress_meas_
                The general stress-measure cycle used by EM algorithms.
    SUB Find_k_res
                Returns the DUT Kelvin resistance.
    SUB Find_res_i_v_
                Returns the DUT resistance, current, and voltage.
    SUB Find_resist_
                Returns the DUT resistance.
    SUB Check_e_monitor
                Checks for extrusion-monitor shorts; used by the EM
                algorithms.
    SUB Heat, Htr_free_
                Support routines for MBLION_SHS.
    SUB Find_res2_, Find_resist2_
                2-terminal DUT resistance
    SUB Find_mpk_res, Find_res_i_v_mp, Find_resist_mp,
                MPSMU equivalents of Find_K_res, Find_res_i_v_,
                Find_resist_.
```

WLRLibhci

```
Routines in WLRLibhci:
  COMMONS:
    COM /Init_idibig/
                Internal storage to save the initial Id, Ib, Ig
                measurements.
  SUBPROGRAMS:
    SUB Rcs_id_28297                                  !File id
    SUB Do_hcis
                Runs through a predetermined hot carrier stress test,
                and stores data in the database by calling HC_LINK_MOS.
    SUB Read_commands
                Builds a hot carrier stress test sequence by reading
                test commands from the user-specified file, and storing
                the results in arrays for use by Do_hcis.
    SUB Measure_par
                Iterates through each specified hot carrier measurement
                test step.
    SUB Do_vt_mos
                Calls VT_MOS and translates input/output for Measure_par.
    SUB Do_idvd_mos
                Calls IDVD_MOS and translates input/output for Measure_par.
    SUB Do_cpv_3_mos
                Calls CPV_3_MOS and translates input/output for
                Measure_par.
    SUB Do_s_mos
                Calls S_MOS and translates input/output for Measure_par.
    SUB Do_dop_mos
                Calls DOP_MOS and translates input/output for Measure_par.
    SUB Do_cpf_3_mos
                Calls CPF_3_MOS and translates input/output for
                Measure_par.
    SUB Do_idibig_mos
                Calls IDIBIG_MOS and translates input/output for
                Measure_par.
```

WLRLibcvanal

```
Routines in WLRLibcvanal:
  COMMONS:
    COM /Cv_calcs/
    COM /Cv_const/
    COM /Cv_data/
                Internal state and CV measurement data commons.
  SUBPROGRAMS:
    SUB Rcs_id_28296                                  !File id
    SUB Hcv_calcs_
                Processes the CV data to return bulk doping, capacitance,
                and flatband, midgap, and threshold parameters.
    SUB Dopc_
                Finds bulk doping as a function of Wmax.
    SUB Cd_
                Calculates the theoretical high-frequency capacitance
                at a specified surface potential.
    SUB Dit_calcs_
                Calculates interface states.
```

WLRLibmath

```
Routines in WLRLibmath:
  SUBPROGRAMS:
    SUB Rcs_id_28298                                    !File id
    SUB Least_square_
                Calculates a least-square fit to a given dataset.
    SUB Calc_vt_
                Analyzes Id and Vg to find Vtext, Vtci, and Gm.
    SUB Lagrange_
                Performs a polynomial interpolation using the
                Lagrange formula.
    SUB Life_calc_
                Extrapolates to find the lifetime fit on linear or
                log scales.
    SUB Calc_s_
                Calculates a subthreshold slope.
```

Data Output Support Libraries

Data output support libraries are used by PDQ-WLR algorithms for all normal data output: database updates, graphics support, and PC-compatible comma-separated value (CSV) files are supported.

The data output support libraries are:

Database libraries:

- WLRdbf_mgmt
- WLR_SPECS_db
- WLR_BASIC_db_template

Graphics and CSV file library:

- WLRgraph_csv

WLRdbf_mgmt

```
Routines in WLRdbf_mgmt:
  COMMONS:
    COM /Pdq_db_dat/
    COM /Pdq_db_dict/
                Commons to pass in data from PDQ-WLR algorithms.
    COM /Pdq_db_main/
                State and control data for the database.
    COM /Pdq_db_wafer/
                Common to pass in wafer/test shell data.
  SUBPROGRAMS:
    SUB Rcs_id_28310                                !File id
    SUB Write_to_db
                Writes a database record, updates record counter.
    SUB Setup_dbf
                Opens existing database file, or creates new data file,
                dictionary, record counter file, and header file.
    SUB Close_dbf
                Closes the current database file.
    SUB Write_rec_num
                Primitive to update the current database record number.
    SUB Parse_path_st
                Primitive to parse user-input path into a fully-qualified
                HP-UX path.
    SUB Pdq_db_off
                Turns off database recording.
    SUB Pdq_db_on
                Turns on database recording.
    SUB Get_db_flag
                Returns a flag to indicate if database recording is
                on or off.
    SUB Get_db_link
                Provides the last written database file name and
                record number.
  FUNCTIONS:
    DEF FNBuild_str$
                Primitive to quote strings.
    DEF FNBuild_real$
                Primitive to translate reals into ASCII strings with six
                significant digits of accuracy, three exponent digits.
    DEF FNBuild_real8$
                Primitive to translate reals into ASCII strings with eight
                significant digits of accuracy, three exponent digits.
    DEF FNBuild_int$
                Primitive to translate integers into ASCII strings.
```

WLR_SPECS_db

```
Routines in WLR_SPECS_db:
  COMMONS:
    COM /Pdq_db_wafer/
                Common to pass out wafer/test shell data.
  SUBPROGRAMS:
    SUB Rcs_id_28309                                !File id
    SUB Pdq_specs_tag
                Returns a requested Agilent SPECS tag variable.
    SUB Pdq_specs_sys
                Returns a requested Agilent SPECS system variable.
    SUB Get_wafer_dat
                Builds the wafer/test shell data including user id,
                date/time, Ids for wafer, module, and device, and the
                module coordinates in microns.
```

WLR_BASIC_db_template

```
Routines in WLR_BASIC_db_template:
  COMMONS:
    COM /Pdq_db_wafer/
               Common to pass out wafer/test shell data.
  SUBPROGRAMS:
    SUB Rcs_id_28307                                  !File id
    SUB Get_wafer_dat
               A programming template to build wafer/test data so
               that the resulting data structures are compatible
               with databases built when running PDQ-WLR under
               Agilent SPECS.
```

WLRgraph_csv

```
Routines in WLRgraph_csv:
  COMMONS:
    COM /Pdq_xgraph_name/
                Common to retain the fully-qualified graph file name
                and file pointer.
  SUBPROGRAMS:
    SUB Rcs_id_28311                                    !File id
    SUB Make_uniq_file
                Builds a unique name from the input request by inserting
                date/time and a version number, as necessary, then opens
                the file.
    SUB Xgraph_open
                Initiates graphics processing, calls Make_uniq_file;
                file suffix is always ".xgr".
    SUB Xgraph_setup
                Sets the graph options.
    SUB Xgraph_data
                Formats the X-Y data for graphing.
    SUB Xgraph_end_plot
                Terminates the current graph, for multiple-graph files.
    SUB Xgraph_close
                Terminates the current graph, closes the graph file,
                and executes the graph command file if so specified.
    SUB Csv_open
                Opens a comma-separated value file using Make_uniq_file;
                reads the "Csv DOS Format" server flag to determine
                record formatting.  File suffix is always ".csv".
    SUB Csv_close
                Closes the current CSV file.
    SUB Plot_csv_decode
                Decodes the Plot$ string input to a PDQ-WLR algorithm.
                Sets flags for graphing and CSV file action; opens
                graph and CSV files as necessary.
    SUB Decode_plotfile
                Primitive used by Plot_csv_decode; manages one file
                at a time.
    SUB Plot_to_file_
                SWORD compatibility routine to decode Plot$ user inputs
                formatted using SWORD instead of PDQ-WLR rules.
```

Agilent IC-MS Compatibility Library

There is only one Agilent IC-MS compatibility library. It provides necessary support functions to PDQ-WLR using Agilent IC-MS function names, and provides functional compatibility between Agilent IC-MS calls and Agilent SPECS or HP BASIC. The library exists only because of the fact that PDQ-WLR is, in part, a derivative of SWORD, which was originally designed to run under Agilent IC-MS. The compatibility is not bi-directional: the library does not make PDQ-WLR compatible with Agilent IC-MS.

```
Routines in ICMSCOMPAT:
  COMMONS:
    COM /Srvrflag/
                Maintains server flags state.
  SUBPROGRAMS:
    SUB Rcs_id_28287                                !File id
    SUB Init_wlr
                Initializes the server flags and parametric tester.
    SUB Srvrflag_load
                Loads the server flag values from the algFlags file.
    SUB Icms_maxtimeout
                No timeouts are supported outside of Agilent IC-MS;
                this routine merely satisfies linkages - does nothing.
    SUB Error_report
                Formats and reports test errors to the user.
    SUB Test_control
                Terminal point for illegal test conditions; stops the
                test by halting HP BASIC.
  FUNCTIONS:
    DEF FNIcms_srvrflag
                Returns the current value of a server flag.
    DEF FNSrvrflag_name$
                Primitive to internally standardize server flag names.
    DEF FNTest_mode
                Returns a constant to indicate that the test system is
                available for use in production mode and engineer mode.
    DEF FNProber_in_use
                Returns a flag to indicate if an autoprober is in use.
```

PDQ-WLR Data Output

PDQ-WLR outputs three types of data:

- Executable graphics files.
- Comma-separated value (CSV) files, suitable for importing into a PC spreadsheet.
- Database entries which contain a complete record of a test, including date/time, DUT information, PDQ-WLR algorithm inputs and outputs, and, as appropriate, pointers to graphics files and CSV files.

Generation of any of these datasets is at the user's discretion. PDQ-WLR maintains an output data hierarchy on the HP-UX /var filesystem in order to manage its output data.

Output Data Hierarchy

PDQ-WLR creates the following areas under the /var filesystem during its installation; all directories are set to rwxrwxrwx permissions.

```
/var/opt/WLR/inputs:
                Used to store the input file for the ALL_HCI sample
                hot carrier program.


/var/opt/WLR/database:
/var/opt/WLR/database/data:
                All PDQ-WLR output data is managed under this path.


/var/opt/WLR/database/data/hci:
/var/opt/WLR/database/data/hci/all_hci:
/var/opt/WLR/database/data/hci/all_hci/lifetime_ptr:
/var/opt/WLR/database/data/hci/all_hci/stress_ptr:
/var/opt/WLR/database/data/hci/stress_tables:
                Used by the ALL_HCI, via HC_LINK_MOS, to associate stress
                data, measurement data, and lifetime data from a hot
                carrier test.


/var/opt/WLR/database/data/measured:
                This is the default area for the PDQ-WLR algorithms'
                database tables.  Each algorithm will create a new
                subdirectory, using its own name, below this path,
                whenever the algorithm is run for the first time with
                the database recording set on.  For example, after the
                first execution of ISO_4_EMR with the default database
                recording conditions set on, the user will find the
                following subdirectory:

        /var/opt/WLR/database/data/measured/Iso_4_emr:


/var/opt/WLR/database/data/csv_files:
                All CSV files are located here, either as actual files,
                or as links to user-specified file names.

/var/opt/WLR/database/data/graphs:
                All executable graphics files are located here, either as
                actual files, or as links to user-specified file names.
```

```
/var/opt/WLR/database/history:
/var/opt/WLR/database/local:
/var/opt/WLR/database/wmap:
/var/opt/WLR/database/config:
/var/opt/WLR/database/data/derived:
/var/opt/WLR/database/data/eps_save:
/var/opt/WLR/database/data/extracted:
/var/opt/WLR/database/data/attrib:
/var/opt/WLR/database/data/plt_save:
/var/opt/WLR/database/data/import:
/var/opt/WLR/database/data/export:
                These areas are all reserved for Agilent PDQ-AT use.
```

Executable Graphic Files

PDQ-WLR algorithms can be commanded to produce graphics outputs in the form of X-Y plots. These graphs serve as a diagnostic tool to verify test system, algorithm, or test structure behavior. Graphics outputs are usually enabled when using a PDQ-WLR algorithm for the first time, when initial tests of a new device type or technology occurs, or whenever there is a question about previous tests and the need arises to re-verify test system integrity or algorithm inputs.

PDQ-WLR produces graphics outputs, if enabled by the user, by writing the data to an HP-UX executable graphics file. The user inputs to PDQ-WLR allows three options:

- Run the executable graphics file as soon as it is produced (at the end of the algorithm test execution);
- save the file for later execution;
- or, both.

When building graphics outputs, PDQ-WLR always creates a unique managed file in /var/opt/WLR/database/data/graphs with the following filename format:

<Algorithm_abbreviation><_date/time<_unique_suffix_number>>.xgr

The date/time string is formatted as YYYY.MM.DD_hh.mm.ss, which is year-2000 compliant with resolution to one second. If a file by this name already exists, then a unique suffix number, beginning at 1, is also appended.

There are three possible dispositions for this file, depending upon user-input criteria:

- If the user only wants an immediate graph, without saving the file, then this executable file is treated as temporary storage and discarded when graphing is complete.

- If the user wishes to save the graphics data, and has elected to let PDQ-WLR manage the graphics data, then the data are permanently stored in the managed file. (This is considered the normal case for users wishing to save an executable graphics file.)
- If the user has elected to save the graphics data under an existing path in the user's writable filespace, then the managed file will be an HP-UX soft-link to the actual file. If the user has selected a filename which already exists in the user's writable filespace, PDQ-WLR will add the date/time/suffix to the file in order to give it a unique name.

Saved executable graphics files can be executed later by simply pointing the HP-UX file manager at /var/opt/WLR/database/graphs and double-clicking the left mouse button on the file of interest. (For most HP-UX configurations, this will leave an additional blank window on screen from the process which executed the graphics file, which the user will have to close after closing the graphics window.) Note that if the user has elected to provide the storage path and filename, and later deletes the graphics file from their filespace, the soft-link in /var/opt/WLR/database/data/graphs pointing to the actual file will become broken, and must also be deleted by the user. Again, the HP-UX file manager is ideal for this, because it graphically displays broken links.

Two server flags in the algFlags file controls graphics behavior when running PDQ-WLR, "Plot On" and "Plot Wait".

The "Plot On" flag is an override for user inputs which specify graphics outputs. If "Plot On" is turned off, then no executable graphics files will be produced. This is a useful utility if one wishes to build test plans which produce graphics to enable exploration during initial testing, and then run the test plan in a production-mode without modifying the test plan, and without producing unwanted graphics.

The "Plot Wait" flag determines the system behavior for the case where graphics are sent to the screen during PDQ-WLR execution. If the "Plot Wait" flag is on, the system will pause until the user closes the xgraph window. If the flag is off, the system will produce the xgraph window on-screen, and proceed to the next test plan step. Care should be taken when using this option; workstations do not have an unlimited amount of X-graphics resources, and can significantly slow down or halt if a large test plan

attempts to simultaneously open a large number of xgraph windows. (The actual number available is dependent upon the system configuration and which other X11 clients are running. In general, more than a dozen xgraph windows on-screen simultaneously is cause for concern.)

Online help for xgraph is included and can be obtained in a terminal window using the command: man xgraph. It may be necessary to set the user's manpath environment variable using the command: export MANPATH=$MANPATH:/usr/local/man. Xgraph allows zooming by selecting a portion of the graph with the left mouse button, which produces a second xgraph window of the selected region. Note that xgraph will produce hardcopy on a Postscript printer, but the printer queue name must be "ps".

At installation, PDQ-WLR deposited the xgraph binary in /usr/local/bin, and the xgraph manpage in /usr/local/man/man1. A sample of X11 user-configurable resources for xgraph are located in /opt/WLR/newconfig/Xgraph, which is distributed for the sake of xgraph completeness, and need not concern the normal PDQ-WLR user.

Comma-Separated Value (CSV) Files

PDQ-WLR algorithms can be commanded to produce tabular listing output files, called CSV files. These files serve as a diagnostic tool to verify test system, algorithm, or test structure behavior. File outputs are usually enabled when using a PDQ-WLR algorithm for the first time, when initial tests of a new device type or technology occurs, or whenever there is a question about previous tests and the need arises to re-verify test system integrity or algorithm inputs.

When building tabular outputs, PDQ-WLR, with the exception of HCI_3_MOS, always creates a unique managed file in /var/opt/WLR/database/data/csv_files with the following filename format:

<Algorithm_abbreviation><_date/time<_unique_suffix_number>>.csv

The date/time string is formatted as YYYY.MM.DD_hh.mm.ss, which is year-2000 compliant with resolution to one second. If a file by this name already exists, then a unique suffix number, beginning at 1, is also appended.

There are two possible dispositions for this file, depending upon user-input criteria:

- If the user wishes to save the tabular data, and has elected to let PDQ-WLR manage the tabular data, then the data are permanently stored in the managed file. (This is considered the normal case for users wishing to save an executable graphics file.)
- If the user has elected to save the tabular data under an existing path in the user's writable filespace, then the managed file will be an HP-UX soft-link to the actual file. If the user has selected a filename which already exists in the user's writable filespace, PDQ-WLR will add the date/time/suffix to the file in order to give it a unique name.

Saved tabular listing files can be viewed later by simply pointing the HP-UX file manager at /var/opt/WLR/database/csv_files and double-clicking the left mouse button on the file of interest.

In general, the tabular listings have the following three sections:

- Algorithm input parameters
- Output parameters
- Measurement data arrays

The first two sections always begin with a "#" sign. Measurement data begins with "# *****", which may include a comment following the asterisks. The next line will be the column titles for the data arrays, followed by the data arrays.

All entries for each line, and the columnar data lines, are separated by commas, and conform to rules for comma-separated value (CSV) files.

CSV files are text files that are directly importable into PC spreadsheet application software, such as Lotus 1-2-3 Release 5 for Windows (Copyright Lotus Development Corporation) or Microsoft Excel 97 (Copyright Microsoft Corporation). Each line of the text file becomes a spreadsheet row, and each item on the text file line becomes a column entry for that row. The items (values) on each line must be separated by commas in order to map correctly to columns.

Here is a CSV file sample from VTDDB_3_CAP:

```
#  5 Aug 1998  15:15:49
# Vtddb_3_cap()  : Input Parameters
# ============================
# Time_max (Sec)     :,  30
# Estress (Mv/cm)    :,  13.25
# Tox (Ang)          :,  75
# Vuse (V)           :,  3.3
# Icomp (A)          :,  .01
# Io (A)             :,  1.E-6
# Testcon 1=acc;2=inv:,  1
# Area (cm^2)        :,  .0001
# Per (cm)           :,  .04
# Tty$ (N/P/Al)      :, P
# Bty$ (N/P)         :, N
# Plot$              :, Y ; systst/2.61/CAP/D6_D3_VTDDB
```

```
# Graph link          :,
# Graph file name     :, ** Graph not saved **
# Link to this file   :, Vtddb_3_1998.08.05_15.15.48.csv
# Name of this file   :, /home/spec23/systst/2.61/CAP/D6_D3_VTDDB.csv
#
# Vtddb_3_cap() : Output Parameters
# ============================
# Tbd (Sec)           :,  1.E+17
# Vbd (V)             :,  1.E+17
# Ibd (A)             :,  1.E+17
# Qbd (C/cm^2)        :,  3.30194375429E-5
# Iuse (Log10 Amps)   :, -10.1375108465
# Fmode               :,  5.1
# Tinit (Sec)         :,  .152954101563
# Iinit (A)           :,  2.06480002403E-10
# Tmax (Sec)          :,  .157806396484
# Imax (A)            :,  2.12599992752E-10
#
# Vtddb_3_cap() : Measurement Data
# ============================
# *****
 Time (Sec),  Curr (A)
 .152954101563, 2.06480002403E-10
 .157806396484, 2.12599992752E-10
 .162567138672, 2.0297999382E-10
 .167358398438, 2.00819993019E-10
 .172149658203, 2.01839995384E-10
 .176940917969, 2.06620001793E-10
```

Note that the input section includes the soft link and filenames for the executable graphics file and a self-reference for the CSV file.

In order to import a PDQ-WLR CSV file, the following conditions must be satisfied:

- The filename must end with ".csv".
- The PC user must choose the "*.csv" File Type in the Open File dialog box.
- Lotus 1-2-3 users need to select the Comma Delimiter under Text Options in the Open File dialog box.
- The CSV file's record formatting must match that expected by the PC application, which differs slightly between Excel and Lotus 1-2-3.

Microsoft Excel 97 reads CSV files in the native HP-UX record format. Lotus 1-2-3 Release 5 reads CSV files with DOS record formatting. The algFlags server flag "Csv DOS Format" controls the CSV files' record formatting. This flag should be on for Lotus users, and off for Excel 97 users. The setting of this flag controls the record formatting when the CSV file is produced by a PDQ-WLR algorithm. Regardless of the original record format, user's can change an HP-UX file to DOS using an HP-UX utility, ux2dos, and change from DOS to HP-UX formatting using another HP-UX utility, dos2ux. The format of one of these utility commands is:

ux2dos < *source_file_name* > *target_file_name*

If there is a doubt about the correct record formatting for a PC application program, the user may create files using an HP-UX text editor, and experiment with file transporting and the ux2dos and dos2ux utilities.

Once the file is in the correct format, it may be transported to a PC via a network or other means; the user should contact their HP-UX system manager for site-specific details.

Database Outputs

PDQ-WLR algorithms create database table files in the /var/opt/WLR/database/data/measured area by default. There is a user-override for using this default area. Below this area, a subdirectory will be created for each PDQ-WLR algorithm which generates database outputs. For example, after the first execution of BEM_3_EMR with the default database recording conditions set on, the user will find the following subdirectory: /var/opt/WLR/database/data/measured/Bem_3_emr. Subdirectories such as this example are called Algorithm Database Areas.

Each Algorithm Database Area will contain the following files:

- One dictionary file, named <algorithm_name>.dict
- One record number counter file, named <algorithm_name>.counter
- One header file, named <algorithm_name>.header
- One or more database table files, whose names will end in ".dat". PDQ-WLR uses the file default.dat in each Algorithm Database Area unless the user overrides the default database settings.

The database table files consist of native HP-UX record-formatted ASCII CSV files. String data are always quoted, and floating-point numbers are always represented with at least six significant digits.

The header file consists of a single record. It contains the database table column titles, in quoted, comma-separated format. By combining a header file with a database table file, and if necessary, filtering the result through ux2dos, and providing an end-result filename ending in ".csv", any database table can be exported to a PC-based spreadsheet application program.

The counter file and dictionary file should never be accessed by the user. The dictionary file consists of entries which show the title and data type for each database table column.

All PDQ-WLR database tables have the same structure:

- Record number
- Date/time
- HP-UX user name of the test system
- Wafer/test shell data consisting of:
  - Product_id
  - Lot_id
  - Wafer_id
  - Wafer_name
  - Die_name
  - Module_name
  - Device_name
  - Module X_position, in microns
  - Module Y_position, in microns
- PDQ-WLR algorithm input parameters
- PDQ-WLR algorithm output parameters, if applicable
- Name of the managed executable graphics file in /var/opt/WLR/database/data/graphs, if applicable
- Name of the managed CSV file in /var/opt/WLR/database/data/csv_files, if applicable

The default for PDQ-WLR is database recording on. The user may turn algorithm recording on and off via algorithms embedded in the WLRdbf_mgmt library, which may accessed in Agilent SPECS using PDQ_DB_OFF_XL and PDQ_DB_ON_XL commands in the test specification.

The user has the option of overriding the Algorithm Database Area root path and of changing the database table name from default.dat to <user_specified>.dat. Changing the default database table name may be a useful strategy for segregating specific tests or test plan executions into smaller, easier-to-find, more manageable files. Because record-numbering is

treated globally within each Algorithm Database Area, small files can be safely combined to create larger files, and large files can be decomposed into smaller ones, without risk of duplicating record numbers.

Changing the root path name for the Algorithm Database Areas is an advanced feature whose use is intended for the most-experienced HP-UX users. By maintaining all data under the /var filesystem, system administration tasks, such as dynamic filesystem re-allocations, and system backups, are kept relatively simple. By maintaining all data under the /var/opt/WLR/database/data/measured area, filespace consumption statistics are easier to obtain, and database table file decomposition and recombination become easier tasks because the record numbers are all managed correctly within the scope of the root path.

Nonetheless, if the user is very comfortable with HP-UX filesystem management and has gained confidence in file manipulations from experience of use with the PDQ-WLR database, Pdq_db_on will let the user change the root path name for the Algorithm Database Areas. If the requested area does not exist, PDQ-WLR will create it, providing that a new entry at the root level (the "/" directory) is not required. In the event of an attempt to create a new root entry, or a path name that PDQ-WLR cannot resolve, PDQ-WLR will automatically revert to its defaults.

Attempts by the user to create or use hidden filenames are ignored; PDQ-WLR will drop the leading period in the filename. Illegal filename characters - / | \ * & - are all converted to underscores.

PDQ-WLR does allow hidden directories and tree-traversal to be performed; so path names may be specified as "../my_directory". Illegal path name characters - | \ * & - are converted to underscores.

Database Failures

If PDQ-WLR cannot create a path you've specified in PDQ_DB_ON or cannot write to said path, the system automatically calls PDQ_DB_OFF internally. This is normal behavior and indicates a file system problem which much be corrected externally.

If you attempt to set a path requiring a new directory to be created in the root path, PDQ-WLR internally resets to the /var/opt/WLR/database/data/measured default path.

3 Managing PDQ-WLR Use

Introduction to Managing PDQ-WLR Use

This section describes the input format for the Plot$ variable found in all PDQ-WLR algorithms which produce graphics outputs and CSV files

The following five data management algorithms are described:

- GET_WLR_FLAG
- PDQ_DB_ON
- PDQ_DB_OFF
- RECORD_DATA
- PLAY_DATA

Data management algorithms do not make DUT connections, produce graphics outputs or CSV files, have parameter outputs, or have compatibility with non-PDQ-WLR products.

Also covered is using thermochucks and the algorithm, SET_TCHUCK.

Additionally, the use of PDQ-WLR with HP BASIC is included in this chapter, customizing and importing algorithms into the Agilent SPECS/PDQ-WLR Library.

PDQ-WLR Algorithm Inputs for Graphics and CSV Files

Each PDQ-WLR algorithm, with the exception of HCI_MOS, is capable of producing graphics outputs and CSV files using an input variable named Plot$. Its format is as follows:

<Graphics Option> < ; <CSV File Option>>

Graphics Options are:

- Y - produce an on-screen plot but do not save the file. Y is a mnemonic for Yes.
- S - save the executable graphics file in /var/opt/WLR/database/data/graphs with a filename selected by PDQ-WLR, but do not produce an on-screen plot. S is a mnemonic for Save.
- B - produce an on-screen plot and use /var/opt/WLR/database/data/graphs to save the file. B is a mnemonic for Both (plot and save).
- path/filename - produces an on-screen plot, saves the file to the user-specified area, and create a soft link to it in /var/opt/WLR/database/data/graphs. When using this Graphics Option, the CSV File Option MUST be specified.
- N or blank - do not produce an on-screen plot or an executable graphics file. N is a mnemonic for No.

CSV File Options are:

- Y or S - save the CSV file to /var/opt/WLR/database/data/csv_files with a filename selected by PDQ-WLR.
- N or blank - do not save a CSV file.
- path/filename - saves the file to the user-specified area, and create a soft link to it in /var/opt/WLR/database/data/csv_files.

If the CSV file option is omitted, then the semicolon field separator may be omitted also; for this case, the CSV file will not be produced. If both options are used, they must be separated by the semicolon field separator; spaces for user readability between the options are allowed and ignored by the

PDQ-WLR input processor.

In order to maintain compatibility for existing SWORD users, the following inputs are recognized:

- "+" translates to "Y ; N".
- "path/filename" only translates to "N ; path/filename".
- "+path/filename" translates to "Y ; path/filename".

The PDQ-WLR Plot$ input format is the preferred method.

GET_WLR_FLAG

Resets the server flags to the values found in the algFlags file. The search order for the algFlags file is the user's local directory area first, followed by the /opt/WLR/etc area.

There are no input or output parameters for this algorithm.

Algorithm Syntax

The syntax for calling this algorithm directly is as follows:

Get_wlr_flag

Theory of Operation

Refer to "The algFlags File" in Chapter 1 for a description of server flags.

PDQ_DB_ON

Activates parameter recording by PDQ-WLR algorithms in the PDQ-WLR database.

The default condition for the database is ***ON***.

Input Variables

Var Name	Var Type	Var Unit	Default Value	Comment/Description
Path$	String	---	"/var/opt/WLR/database/data/measured"	Root path for the Algorithm Database Areas
File$	String	---	"default.dat"	Data table name to use within the Algorithm Database Areas

Algorithm Syntax

The syntax for calling the algorithm directly is as follows:

Pdq_db_on (Path$,File$)

Theory of Operation

Use of the Path$ input is discouraged for normal use conditions. Use the File$ input, with Path$ blank, to manage your test data using a file naming convention relevant to your testing practices. Making multiple PDQ_DB_ON calls in the same test plan without also calling PDQ_DB_OFF is an acceptable practice. This technique can be used to create multiple datasets within a single test plan with a minimum of complexity. Refer to "Database Outputs" in Chapter 2.

PDQ_DB_OFF

Deactivates parameter recording by PDQ-WLR algorithms in the PDQ-WLR database.

The default condition for the database is ***ON***.

There are no input or output parameters for this algorithm.

Algorithm Syntax

The syntax for calling the algorithm directly is as follows:

Pdq_db_off

Theory of Operation

Refer to "Database Outputs" in Chapter 2.

RECORD_DATA

Activates measurement recording by PDQ-WLR algorithms using the Agilent 4062UX TIS; does not apply to the Agilent 4070 series.

Input Variables

Var Name	Var Type	Var Unit	Default Value	Range	Comment/Description
Filename$	String	---	""		File to store TIS recording data.

Algorithm Syntax

The syntax for calling the algorithm directly is as follows:

Record_data (Filename$)

Theory of Operation

Refer to the Agilent 4062UX TIS documentation. The Agilent 4062UX is capable of mirroring the measurement outputs returned to an algorithm into a user-defined file for later playback. This algorithm is of limited use to the typical PDQ-WLR user, but may provide some benefit for baselining algorithm performance.

PLAY_DATA

Activates measurement recording by PDQ-WLR algorithms using the Agilent 4062UX TIS; does not apply to the Agilent 4070 series.

Input Variables

Var Name	Var Type	Var Unit	Default Value	Range	Comment/Description
Filename$	String	---	""		File to read recorded TIS data.

Algorithm Syntax

The syntax for calling the algorithm directly is as follows:

Play_data (Filename$)

Theory of Operation

Refer to the Agilent 4062UX TIS documentation. The Agilent 4062UX is capable of mirroring the measurement outputs returned to an algorithm into a user-defined file for later playback. This algorithm is of limited use to the typical PDQ-WLR user, but may provide some benefit for baselining algorithm performance.

Note of Using Thermochucks

PDQ-WLR includes support for a number of thermochucks. The currently supported list is found in the /opt/WLR/etc/WLR.config file, which is updated from time to time when new thermochuck driver software is added to PDQ-WLR by update releases (patches).

In the WLR.config file, you will need to uncomment the INSTR reference for your thermochuck.

There is a special driver for Agilent SPECS users called SPECS_CHUCK. This driver software is used when your Agilent SPECS Prober Library already includes the necessary thermochuck driver software. When using this device driver, be sure to edit the WLR.config entries to reflect the minimum and maximum temperature range for your particular thermochuck.

Most thermochucks do not include an option to turn them off via software command. When using a thermochuck, the best that can be done via software is to lower it to its minimum setting.

Operators must be trained to manually turn off the thermochuck when PDQ-WLR testing is complete.

SET_TCHUCK

There are times when you wish to establish a particular temperature for the entire wafer before performing any tests. The SET_TCHUCK algorithm allows you to do exactly that.

The thermochuck must be defined in WLR.config to use this algorithm.

Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Temperature	Real	°C	25	LOWER_TEMP - UPPER_TEMP values in WLR.config	Desired chuck temperature
Soak_time	Real	sec		0 ~ 1000	Temperature stabilization time

Device Variables

Var Name	Var Type	Comment/Description
None		

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
T_error	String		Error return: null string = no error

Algorithm Syntax

The syntax to call the algorithm directly is as follows:

Set_tchuck (Temperature, Soak_time, T_error$)

Using PDQ-WLR with HP BASIC

The following sections explains the use of PDQ-WLR with HP BASIC.

Preparing WLR_BASIC_db_template

The PDQ Database, as mentioned earlier, has entries for all tests consisting of three parts:

data from the test shell
test algorithm inputs
test algorithm outputs

The data from the test shell is provided through a database library software layer. A template for this layer is provided at /opt/WLR/lib/WLR_BASIC_db_template. This template, as supplied, is structured to create test shell data that is completely compatible with Agilent SPECS. By following this template, you can provide database outputs by HP BASIC programs which can be freely interchanged with Agilent SPECS.

The recommended action for the user is to:

1. Edit the WLR_BASIC_db_template to correctly capture the needed test variable information.
2. Save the new file under another name, such as /opt/WLR/lib/WLR_BASIC_db.
3. Update the WLR.config file to use this layer.
4. Perform your software build using *makewlrprog*.

The WLR_BASIC_db_template, as provided, will operate without errors, but will provide blanks and zeros for the critical test parameters.

If the test data available for your custom software effort cannot be made to emulate the Agilent SPECS structure provided for you in the WLR_BASIC_db_template, you can still successfully store and use your data. In this event **it is strongly advised that you use the PDQ_DB_ON algorithm to set a different path (other than the default) for your test sessions**.

You should also do this whenever you change the WLR_BASIC_db_template. In this way, your system can support multiple definitions of the your critical test data. For example, if you have one definition of WLR_BASIC_db and use ISO_4_EMR, then PDQ-WLR will create database outputs at /opt/WLR/lib/Iso_4_emr. Included in these outputs will be the dictionary defintion for all variables in the database for that test run. By creating a new defintion of WLR_BASIC_db, your old dictionary will not be correct, and PDQ-WLR will have no way of knowing that this is what you have done; it will

write the data out as instructed, but you will experience data retrieval problems later. This problem is completely solved by changing the path input to PDQ_DB_ON.

Preparing WLR_BASIC_Prober_template

See Prober Interface Libraries in Chapter 2.

Calling Algorithms Using HP BASIC

This section explains how to execute the WLR algorithms outside of Agilent's test shells. Although users receive maximum benefit by using the algorithms under Agilent SPECS, the algorithms can be executed from within a user-written HP BASIC program.

Device Specifications

When using Agilent SPECS, fixed characteristics such as the type of terminal (pin) connections and device parameters (e.g., area of a capacitor) must be defined for each device type. For HP BASIC main programs, the device data will always be placed into the ***Pins*** array for input into a PDQ-WLR algorithm.

Sample CAP Device Specification

Terminal Connections	Position in Pins Array	Description
Top	1	Top electrode
Bot	2	Bottom electrode
Sub	3	Substrate contact
Chuck	4	Wafer chuck

Device Parameters	Data Type	Description
Area	Real	cm^2, Area of active region
Per	Real	cm, Perimeter of active region
Tty$	String	(N/P/Al), Top electrode type
Bty$	String	(N/P), Dopant type of bulk region

Algorithm Parameter Format

The calling format for PDQ-WLR algorithms has the following fixed order:

- Input parameters, followed by
- Pins array (terminal connections), followed by

- Device parameters, followed by
- Output parameters

This format must be followed in order to call a PDQ-WLR algorithm. The device definitions are generic to handle a variety of applications, and a PDQ-WLR algorithm may not use all the possible device parameters. This may seem like unneeded information for those cases, but this approach to argument-list management eases development constraints when designing software to be compatible with a variety of other systems.

Determining Calling Parameters

The device type and the parameters are discussed in the introductory section before each WLR category (Oxide, Electromigration, and Hot Carrier). The input and output parameters are defined in the algorithm documentation section.

In addition to the printed documentation, the source code contains a full discussion of all the parameters. The algorithm code can be found in the **/opt/WLR/BASIC** directory.

Making a PDQ-WLR Program

For most users, creating a main program directly in the HP BASIC window is easiest; it provides on-the-fly syntax checking. You may also create HP BASIC programs as ASCII text files using a UNIX text editor. Either way, the program will have to provide the necessary input parameters, Pins array, and device parameters for each PDQ-WLR algorithm you intend to call. The argument list will also have to receive the output parameters. *There are NO optional arguments in any of the PDQ-WLR algorithms.* The argument list for each algorithm is described in its appropriate section of Chapters 4, 5, or 6. If in doubt, you may always open the algorithm from /opt/WLR/BASIC using HP BASIC, and review it directly.

The utility *makewlrprog* attaches the necessary PDQ-WLR routines to the main program to create a PDQ-WLR program. This utility links in the PDQ-WLR support libraries and provides the necessary external equipment drivers, as defined in the /opt/WLR/etc/WLR.config file. Refer to Chapter 1, **Installation**, for details on using *makewlrprog* to build PDQ-WLR main programs.

PDQ-WLR and HP BASIC

PDQ-WLR is a very large software system. The recommeded workspace is 16 MB, which can be specified on the HP-UX command line to start HP BASIC: *rmb -16M*.

Custom Algorithms and the PDQ Database

On occasion, you may wish to add your own custom test algorithms to PDQ-WLR and then have your software send outputs to the PDQ Database.

For Agilent SPECS users, this is a very easy task, requiring two steps:

add your algorithm using the Agilent SPECS Builder utility, using the current PDQ-WLR library as the target

execute /opt/SPECS/usr/bin/ExtendPDQLib

The ExtendPDQLib will ask for source algorithm library to operate on. In this case, specify the current PDQ-WLR library as your input. The ExtendPDQLib tool will then auto-detect your new algorithm and ask you if you wish to have a database software thin layer built for it. When you agree, ExtendPDQLib will create a new algorithm, called <your_original_algorithm_name>_DB, and place it in the library. Your original custom algorithm will not be modified in any way.

ExtendPDQLib will create an algorithm that takes two simple steps:

call your algorithm

send test data, algorithm inputs and outputs to the PDQ Database

Call the _DB version of your algorithm whenever you wish to send data to the PDQ Database.

For HP BASIC users writing their own custom software, it's not quite so simple. In this case, you must perform the database call yourself. To do this, you will need to include the PDQ Database shared common blocks, fill the variables in these common blocks in exactly the same way that a PDQ-WLR algorithm does, give yourself a unique entry name (the algorithm name is best), and call the Write_to_db database routine. The best thing to do is to look at the embedded Database subroutine in one of the existing PDQ-WLR algorithms, such as VRAMP_3_CAP, and copy what it does; remember to change the name to your algorithm on the Write_to_db call.

Importing Algorithms into the Library

Agilent SPECS versions 2.4 and above provide an easy facility for this. PDQ-WLR also provides this capability with the tool /opt/SPECS/usr/bin/ExtendPDQLib. The ExtendPDQLib tool can be used to simultaneously provide database versions of your algorithms during the import. (See "Custom Algorithms and the PDQ Database" above.)

If you have upgraded to Agilent SPECS 2.4 or later, your previous algorithm libraries at located in /opt/SPECS/usr/alg. Any libraries you wish to use with ExtendPDQLib will have to be moved to /opt/SPECS/usr/alg/measure.

Testing Your Installation Under Agilent SPECS

PDQ-WLR includes a new routine, called PDQ_TEST_XL, that can be used to test your integration of PDQ-WLR with Agilent SPECS. This routine accepts three dummy inputs, and copies the input values to three outputs. It produces a simple xgraph, and writes its input and output data to the /var/opt/WLR/database/data/csv_files area in the form of a CSV file. No DUT connections are made. This routine does produce database outputs.

4 Oxide Integrity

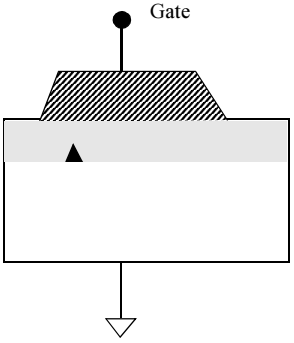
Introduction to Oxide Integrity (OX)

This section provides a general overview of the WLR Oxide Integrity algorithms.

Voltage across oxide causes current to flow through oxide. Small defects in oxides or silicon will cause more current.

Current causes damage in oxide.

Critical damage causes thermal breakdown. Defects breakdown at lower voltage.



Oxide Integrity Algorithms

Name	Device	Comment/Description
VRAMP_3	CAP	JEDEC JESD35 compliant voltage ramp with intelligent failure analysis
VTDDB_3	CAP	Constant voltage time dependent dielectric breakdown
JRAMP_3	CAP	JEDEC JESD35 current ramp
JTDDB_3	CAP	Constant current time dependent dielectric breakdown
HFCV_CORR_3	CAP	Correction routine for HFCV_3
HFCV_3	CAP	High frequency CV curve and extract physical parameters.
QCV_3	CAP	Low frequency CV using quasi-static technique
CV_3	CAP	HFCV_3 and QCV_3 with interface state analysis
MBLION_3	CAP	Mobile ion using triangular voltage sweep technique and CAP
MBLION_SHS	SHS	Mobile ion with on-chip self-heated structure

CAP Test Structure Definition

The Oxide algorithms are defined for the device type CAP (except for **MBLION_SHS**). This device type can include capacitor or MOSFET insulating structures with the a top and bottom electrode. The optional substrate and/or chuck contacts can be used to connect to a minority carrier contact for breakdown in inversion.

CAP Device Specifications

Terminal Connections	Position in Pins Array	Comment/Description
Top	1	Top electrode
Bot	2	Bottom electrode
Sub	3	Substrate contact
Chuck	4	Wafer chuck

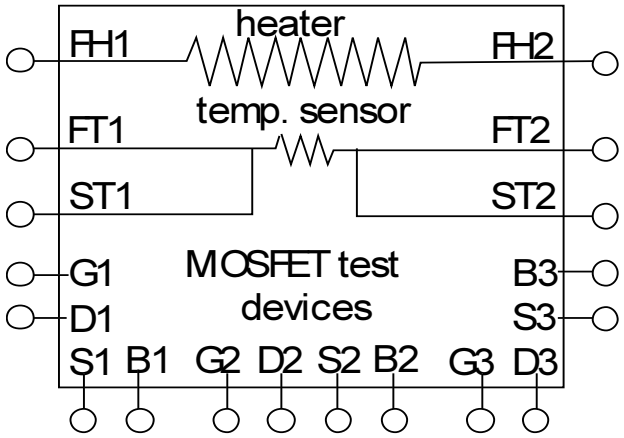
Device Parameters	Data Type	Comment/Description
Area	Real	cm^2, Area of active region
Per†	Real	cm, Perimeter of active region
Tty$†	String	(N/P/Al), Top electrode type
Bty$	String	(N/P), Dopant type of bulk region

† Not used by WLR Oxide Algorithms

Self-Heated Test Structure Definition

The **MBLION_SHS** algorithm uses a Self-Heated Structure. Additional information may be found in **MBLION_SHS**.

Self-Heated Device Schematic



Self-Heated Structure Device Specification

Terminal Connections	Position in Pins Array	Comment/Description	
Fh1	1	High side of Heater	Heater
Fh2	2	Low side of Heater	
St1	3	High Sense of Temperature Sensor	Thermo-meter
St2	4	Low Sense of Temperature Sensor	
Ft1	5	High Force of Temperature Sensor	
Ft2	6	Low Force of Temperature Sensor	
Mg2	7	Optional Gate Bias during Stress*	
Mg3	8	Optional Gate Bias during Stress*	
G1	9	Gate MOSFET Device 1 (optional)**	Device 1
D1	10	Drain Device 1	
S1	11	Source Device 1	
B1	12	Bulk Device 1	
Sub1	13	Substrate Contact Device 1***	
Chk1	14	Wafer Chuck Device 1***	
G2	15	Gate MOSFET Device 2 (optional)**	Device 2
D2	16	Drain Device 2	
S2	17	Source Device 2	
B2	18	Bulk Device 2	
Sub2	19	Substrate Contact Device 2***	
Chk2	20	Wafer Chuck Device 2***	
G3	21	Gate MOSFET Device 3 (optional)**	Device 3
D3	22	Drain Device 3	
S3	23	Source Device 3	
B3	24	Bulk Device 3	
Sub3	25	Substrate Contact Device 3***	
Chk3	26	Wafer Chuck Device 3***	

*Mg2 and Mg3 provide pins for the gate-to-substrate voltage to be applied during the stress.

**If not defined (i.e. set equal to 0), then algorithm will not make threshold voltage measurements.

***Not necessary to define.

Theory of Operation

The Oxide Breakdown algorithms force a charge through an oxide until failure. The Voltage Ramp (**VRAMP_3_CAP)** and the Current Ramp (**JRAMP_CAP**) follow the Joint Electron Device Engineering Council (JEDEC) standard JESD35. **VRAMP_3_CAP** is usually more sensitive at categorizing defects than **JRAMP_3_CAP**. Both these methods are fairly quick. The JEDEC documentation details the restrictions and limitations of these two tests. The **VRAMP_3_CAP** algorithm follows the JEDEC specification in the application of the voltage ramp and uses intelligent failure analysis to determine the breakdown with four empirical methods. These methods have been used over the last seven years to accurately characterize thin oxides in development and in production.

Currently JEDEC has not released a standard for the Time Dependent Dielectric Breakdown (TDDB) measurement. The routine **VTDDB_3_CAP** performs a constant (non-ramped) voltage stress. The **JTDDB_3_CAP** routine performs a constant current oxide breakdown test.

The CV characterization routines (**HFCV_3_CAP, QCV_3_CAP** and **CV_3_CAP**) are used to determine fundamental MOS system parameters such as oxide thickness, flatband voltage, midgap voltage, substrate doping and interface state density. These routines can also be used to determine the amount of charge trapping and interface states resulting from accelerated oxide stressing. More realistically, interface states and trapped charge during an oxide test are easier and faster to measure using **CPF_3_MOS** or **CPV_3_MOS**.

MBLION_SHS and **MBLION_3_CAP** are useful for characterizing and identifying mobile ions introduced into the integrating circuit manufacturing process. This mobile charge in oxides is primarily impurities sodium (Na+), potassium (K+), or lithium (Li+). **MBLION_SHS** uses a special self-heated on-chip structure to perform fast, high-temperature biased temperature stress without a hot chuck. **MBLION_3_CAP** performs a triangular voltage stress with the help of a hot chuck.

CAP Test Structures Design Rules

The following paragraphs present six basic rules for the design of oxide test structures to obtain predictive results:

1) **Minimize Conductor Length:** The conductor running to the gates and well/substrate/tub must be of a minimum length in order to minimize series resistance to the test structure. If it is not, there can be a significant voltage drop in this line, especially at or near the currents used for rapid breakdown testing. If such a voltage drop should occur, large erroneous Vbd readings will result: long lines of high resistance may also cause the opening of the line by fusing before the oxide test structure fails. This can lead to either an absence of data or very large values of Vbd that may not be interpreted for what they are - an open circuit.

2) **Maximize Conductor Width:** All statements made pertaining to rule one also apply here. Many designers will naturally wish to use a standard minimum geometry line when laying out oxide test structures. It is important to maximize the line width, given the constraints imposed by other nearby structures.

3) **Maximize Number of Gate Contacts:** By maximizing the number of gate contacts, the structure will have a minimum series contact resistance, thereby ensuring that nearly all of the stress voltage is dropped across the oxide. With one or only a few contacts, we have observed contact failure before or during oxide breakdown. Such failures can result in abnormally large values of Vbd recorded, as well as mimicking a self-healing failure.

4) **Maximize Number of Well/Substrate/Tub Contacts:** All statements made pertaining to rule three apply here.

5) **Provide a Minority Carrier Source:** Most oxide breakdown measurements are conducted in accumulation, where there is no need for a minority carrier source. However, in the event that the oxides are ever to be biased in depletion or inversion, a source of minority carriers should be provided so that the results are not carrier limited. Use of light, although potentially appropriate for smaller capacitor structures, will not provide the needed number of minority carriers in

the center of the larger gate area structures. By providing a permanent source of minority carriers, you can ensure the correct operation of the capacitors over both bias polarities.

6) **Design Capacitors to Have Representative Areas:** The major purpose of capacitor test structures is to quantify the defect density of the gate oxide. It is important therefore to have a structure with an area equal to the total gate area of the technology in question. In the case of measuring the "intrinsic" breakdown field of gate oxides, it has been shown that the field varies as 1/log(Area) according to the extreme value distribution. Very small capacitors may lead to erroneously large values of Vbd. The best sets of oxide test structures will have a series of capacitor areas, varying from the area of a single transistor gate to the area of all of the gates combined.

Applicable Test Structures

For users of PDQ-FAB WLR test structures, a separate product available from Agilent, the following table lists the PDQ-WLR software algorithm recommended for use with the test structure. Other algorithms may be used such as **VTDDB__3_MOS** or **JTDDB_3_MOS**.

PDQ-FAB WLR Oxide Integrity Test Structures

Structure	Structure Name	Primary Applicable WLR Algorithms
Oxide Breakdown: Logarithmically scaled area gate oxide caps for fast oxide defect detection	CAP_A_N CAP_A_P	VRAMP_3 JRAMP_3
Oxide Breakdown: Implant Edge and Field Oxide Edge Intensive caps, Full topography consisting of 100s of parallel MOSFETS	CAP_FIA_N CAP_FIA_P CAP_TRANS_N CAP_TRANS_P	VRAMP_3 JRAMP_3
Oxide Breakdown: Interlevel Dielectric Integrity Defect Detection: Area and Edge Intensive caps	CAP_ID_F, CAP_ID_FW CAP_ID_N[1] CAP_ID_NE[1]	VRAMP_3 JRAMP_3

Mobile Ion Contamination	CAP_A_N CAP_A_P	MBLION_3
Self-Heated Mobile Ion	MBL_1, MBL_2	MBLION_SHS

[1] N here is number of interlevel dielectric layers

The next table lists some of the possible oxide-related failure mechanisms detected using the PDQ-FAB WLR test structures and software.

Oxide Integrity Test Structures and Failure Mechanisms

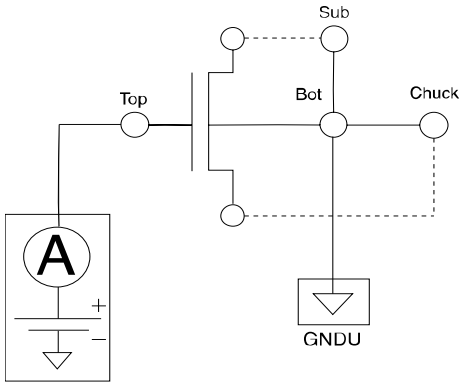
Test Structure Description	Failure Mechanisms	Tests
Oxide Integrity (Gate, Field, Implant Edge)	Heavy metal contamination (Fe, Al); Si Defects; Oxygen Precipitates; Particles; Stress at oxide; Kooi effect; S/D undercut	V Ramp, J Ramp, TDDB, C-V Characterization
Mobile Ions	Ion Contamination in Oxide	Temperature Bias

VRAMP_3_CAP

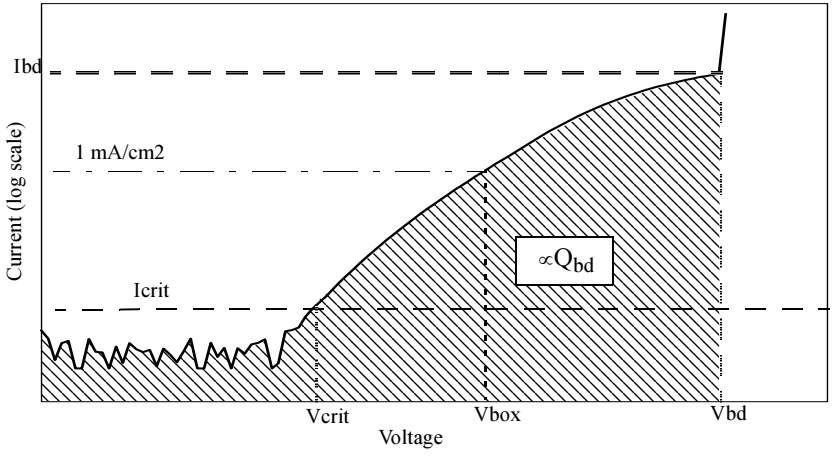
Oxide breakdown with voltage ramp in compliance JEDEC Standard JESD35.

Voltage is increased in linear time steps until the oxide ruptures or the ramp completes. Breakdown is determined by 4 different techniques, which is necessary for breakdown detection in thin oxides. The current and step time is measured at each step and recorded to insure accurate calculation of accumulated charge per area (Qbd). Capacitor or MOSFET structures can both be used in accumulation or inversion. Optional connection(s) for minority carrier contact are shown by dashed lines for breakdown in inversion.

Test Configuration



Electrical Characteristics



Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Vstart†	Real	Volts	3.3	0.1~20	Start gate voltage
Vmax†	Real	Volts	19	1~200	Ending gate voltage
Vstep	Real	Volts	0.075	0.01~5	Gate bias step size
Tox	Real	Å	75	10~10E3	Design oxide thickness
Vuse†	Real	Volts	3.3	0.1~20	Post test check voltage
Icomp	Real	Amps	1.0E-1	1.0E-9~0.35	Compliance limit of sweep
Icrit	Real	Amps	1.0E-10	1.0E-15~1.0E-2	Lowest current level of search
Ramp_rate	Real	MV/cm•sec	1.0	0.01~15.0	Electric field ramp rate
Io	Real	Amps	1.0E-6	1.0E-9~1.0E-3	Pre/Post test leakage current
Testcon	Integer		1	1~2	Test in: 1=accumulation, 2=inversion
Plot$	String	——	"Y"††		Plot/Save Data

† Must be specified as a positive number. The algorithm will determine the correct polarities based on the bulk type (N or P) and test condition (accumulation or inversion).

††A "Y" plots current versus voltage on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files*.

Device Variables*

Var Name	Var Type	Comment/Description
Area	Real	cm^2, Area of active region
Per†	Real	cm, Perimeter of active region
Tty$†	String	(N/P/Al), Top electrode type
Bty$	String	(N/P), Dopant type of bulk region

† Not used by WLR Oxide Algorithms

*These are the input variables for the CAP Device. A properly designed transistor structure may also be used provided the gate, bulk pins are correctly defined. Note that a PMOS (NMOS) transistor structure has a N (P) Bty$.

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Vbd	Real	Volts	Breakdown voltage
Ebd	Real	MV/cm	Breakdown field
Qbd	Real	C/cm^2	Accumulated charge
Fmode	Real	——	Failure mode 1 - Failed initial test 2.X - Catastrophic breakdown 3.Y - Masked catastrophic breakdown 4.X - Non-catastrophic breakdown 5.Y - Other X=1,2,3,4 is breakdown method Y=1,2 is test stop condition
Iuse	Real	Log_{10} Amp	Current at pre-test Vuse
Ibd	Real	Log_{10} Amp	Current at Vbd
Vcrit	Real	Volts	Voltage at Icrit
Vbox	Real	Volts	Voltage at 1 mA/cm^2

Algorithm Syntax

The syntax for calling the algorithm directly (that is, outside of Agilent SPECS) is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in *Introduction to Oxide Integrity* section.

Vramp_3_cap(Vstart, Vmax, Vstep, Tox, Vuse, Icomp, Icrit, Ramp_rate, Io,
Testcon, Plot$,
Pins(*), Area, Per, Tty$, Bty$,
Vbd, Ebd, Qbd, Fmode, Iuse, Ibd, Vcrit, Vbox)

Theory of Operation

The *Introduction to Oxide Integrity* section provides a general overview.

VRAMP_3_CAP measures the oxide breakdown voltage (or electric field) for an insulator by increasing the applied voltage in uniform steps until the oxide breaks down. As the voltage across the oxide increases, the electron current flowing through the oxide increases due to Fowler-Nordheim (or Direct) tunneling. During this process holes are also generated in the oxide. Since these holes are fairly immobile in SiO_2, they cause significant damage to the oxide. One explanation for oxide breakdown is that these holes (and electrons) cause an increase in neutral electron traps in the oxide [1]. If enough of these traps are generated to form a conductive path of traps between the gate and substrate, this leads to a large electron current and thermal breakdown.

[1] R. Degraeve, et.al., "New insights in the relation between electron trap generation and the statistical properties of oxide breakdown," IEEE Trans. Electron. Dev., vol. 45, pp. 904-911, April 1998.

Methodology

The methodology for performing a voltage ramp oxide breakdown is described in JEDEC standard JESD35. First, a pre-ramp test is performed. Vuse is applied across the capacitor and the current Iuse is measured. If Iuse is greater than Io, then the capacitor is considered to be an initial failure. If Iuse is less than Io, then the voltage is ramped from Vstart in uniform Vsteps until oxide breakdown is detected or Vmax (or a maximum Qbd) is reached. A post ramp test (by again applying Vuse) is then performed to verify the oxide breakdown.

During a **VRAMP_3_CAP** test, the current and voltage is analyzed in real time to determine the voltage right before breakdown (Vbd). This real time analysis saves overall test time. The algorithm also calculates the electric field just before breakdown (Ebd). Ebd is calculated according in accordance with JESD35 by dividing Vbd by Tox.

VRAMP_3_CAP also stores the current (Ibd) at the breakdown voltage (Vbd) and the leakage current (Iuse) at use voltage (Vuse). The I-V curve is further parameterized by Vcrit and Vbox. Vcrit is the voltage at Icrit. Vbox is the voltage where the current through the device is 1 mA/cm^2. Both of these voltages are determined by interpolation of the I-V data.

At each voltage step, the algorithm multiplies the measured current by the measured step time to calculate the charge. The charge at each step is added together to determine integrated charge density to breakdown (Qbd).

The **VRAMP_3_CAP** algorithm determines the failure type (or mode) and outputs the breakdown detection method used. The failures modes are classified into five modes according to how they respond to the initial pre-ramp test, the ramp test and a post-ramp test. These failure modes are defined in the table below.

Failure Mode (Fmode)	Initial Test	Ramp Test	Post Test
1-Initial	Fail	N/A	N/A
2.X-Catastrophic†	Pass	Fail	Fail
3.Y-Masked Catastrophic††	Pass	Pass	Fail
4.X-Non Catastrophic†	Pass	Fail	Pass
5.Y-Others††	Pass	Pass	Pass

† X is 1,2,3 or 4 depending on the breakdown detection method used.

†† Y is 1 or 2 depending on test stop condition.

With properly designed test structures, most capacitors will break down catastrophically. They break down during the voltage ramp and remain shorted between the gate and the substrate (Fmode = 2.X). Sometimes, however, the energy dissipated in the breakdown is enough to explosively clear the short, leading to an unshorted gate and a non-catastrophic failure (Fmode = 4.X). The masked catastrophic (Fmode = 3.Y) appears to pass the ramp test, but then doesn't work post test. This type of failure will occur if the capacitor breaks down in a manner which is not detected during the ramp. A failure mode 5.Y indicates that the capacitor did not break down. The "Y" in these last two modes indicates how the test was stopped. If Y=1, then the voltage ramp finished at Vmax. This may indicate an open circuit due to test fixture or test structure problems. If Y=2, then the test was stopped because Qbd exceeded 50 C/cm^2. JESD35 recommends a maximum Qbd (Qmax) of 50 C/cm^2. A Y=2 condition may indicate a test fixture or test system problem which results in a large leakage current.

For thin oxides, that don't breakdown into a "hard" short, a number of empirical methods to detect the breakdown were developed. Most of these criteria are now part of the JEDEC Standard JESD35. These breakdown detection techniques also work for thicker oxides.

The breakdown detection method is represented by the "X" in Fmode 2.X and Fmode 4.X in the previous table. **VRAMP_3_CAP** uses four different methods to detect breakdown. These detection methods are:

1) **COMPLIANCE METHOD:** If the measured current is within 1% of Icomp then the corresponding voltage at that step is flagged as a voltage breakdown point. (Must have occurred above Icrit).

2) **10x METHOD:** If the measured current is 10 times greater than the previous measured current then the voltage of the previous step is flagged as a voltage breakdown point. (Must have occurred above Icrit).

3) **SLOPE METHOD:** A slope change of 2.5x or more in the current (log scale)-voltage (linear scale). The magnitude of the current must have also changed by at least 2x. (These criteria must be met above Icrit to qualify as a slope breakdown).

4) **1000x METHOD:** If the measured current at a step is 1000 times greater than the previous measured current then the voltage of the previous step is flagged as a voltage breakdown point. (This is the only check which will occur above and below Icrit).

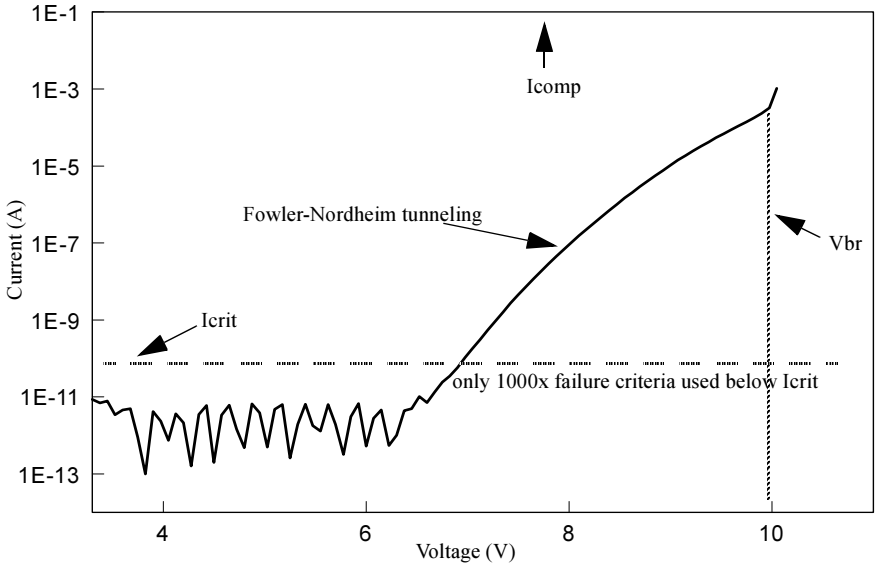
If multiple methods detect breakdown at the same voltage, then only the lowest detection method is reported. For example, if both the compliance method and 10x method report the same breakdown voltage, the breakdown detection method reported is 1 (e.g. 2.1 or 4.1).

The breakdown detection method for failure modes 2.X and 4.X are summarized in the following table.

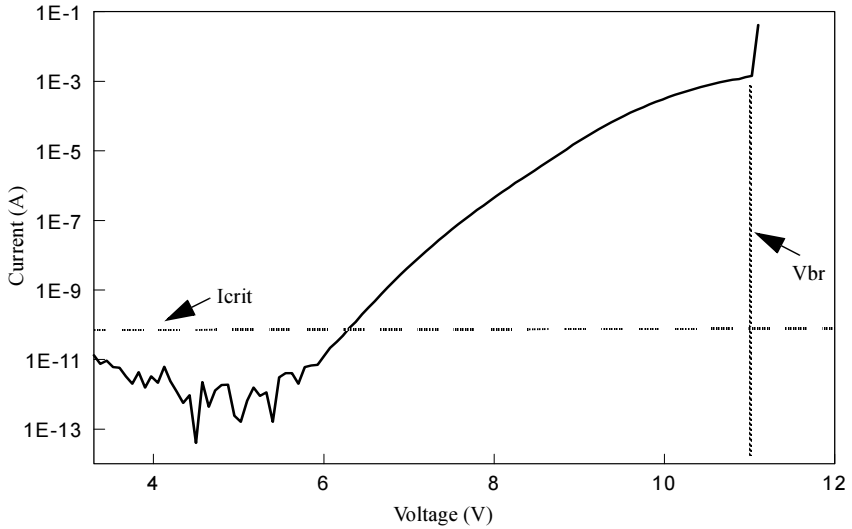
Breakdown Detection Method	Failure Mode 2 (Catastrophic)	Failure Mode 4 (Non-Catastrophic)
1 - Compliance	2.1	4.1
2 - 10x	2.2	4.2
3 - Slope	2.3	4.3
4 - 1000x	2.4	4.4

We will illustrate the last 3 breakdown detection methods with real examples. The first figure shows a characteristic breakdown of a capacitor test structure with an oxide thickness of 75 Angstroms. During the early part of the ramp, only noise is observed (in this case due to probe card and fixture leakage). At around 6.7 Volts, Fowler-Nordheim tunneling is larger than the

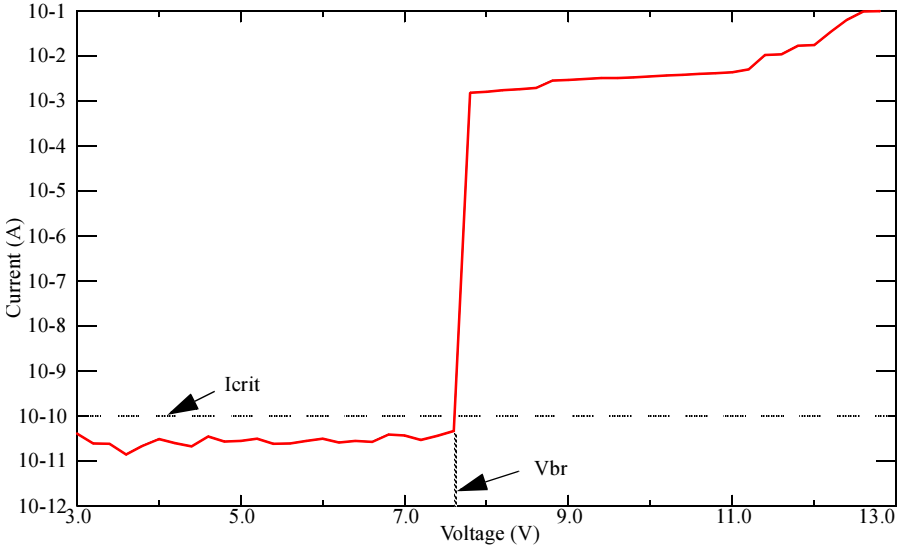
noise. This tunneling current increases with increasing voltage until breakdown at 9.975 Volts. This breakdown is detected by the algorithm as a subtle change in slope (Failure mode = 2.3).



The next graph shows **VRAMP_3_CAP** detecting a 10x-check type failure (Failure mode = 2.2).



Sometimes a capacitor will break down out of the noise. This is shown in the figure below. Here the algorithm defines the breakdown as a 1000x change in current. The failure mode is 2.4.



Testing notes: Due to the many possible user combinations of Vstart, Vmax and Ramp_rate, the step time may be smaller than a minimum time. In this case, the code will abort the test rather than continue. In order to avoid this possibility, the user can calculate the step time using the following:

Step_time = (Vmax - Vstart) / Vpersec / Steps > Min. time

where Vpersec = Ramp_rate * Tox * 1.0E-2
Steps = INT ((Vmax - Vstart) / Vstep) + 1)

The minimum time (Min. time) in the algorithm is set to 10ms. This is a conservative estimate for algorithm controlling a Agilent 4070 series test system. This 10ms is a reasonable estimate for algorithm controlling a Agilent 4062UX. In any event, the actual step time is measured at each voltage step and is output with the I-V plot data.

The JEDEC requirement for voltage ramp requires a step_time of 100ms with a Ramp_rate of 1 MV/cm*sec. Vstep should also be set so that equivalent field step is 0.1 MV/cm (e.g. Vstep=0.05 for a 50 Angstrom gate oxide).

For currents approximately less than 1 nA and fast (non-JEDEC) ramp rates, the step time may be longer than specified with the Ramp_rate and Vstep values. This is due to the time necessary to accurately measure low currents.

In general, the measurement time in this current range is less for the Agilent 4070 series than for the Agilent 4062UX. This does not effect the output parameters as the Qbd is calculated based on the measured step time and current.

Because of UNIX operating system scheduling, the user may occasionally observer a longer than expected step time for one data point during a ramp. This is a function of the UNIX operating system and will occur much less than 1% of the time. This will not impact the integrity of the output variables.

Data File Format: When using the Plot$ variable to output ASCII data to a file the resulting format will be:

Input parameters		
Output parameters		
Step_time (secs)	Current (A)	Voltage (V)
<data>	<data>	<data>

Test Structure

See "Test Structures" in the *Introduction to Oxide Integrity* section.

Step-by-Step Action

1) Check for legal input values. If any are out of range, output variables are set to flag values 2.0E+30 and algorithm exits.
2) Connect SMUs as in Test Configuration diagram.
3) Determine polarity based on substrate type and test condition (accumulation or inversion).
4) Calculate step time to produce user specified Ramp_rate. If too small, set output values to 2.1E+30, exit algorithm.
5) Perform pre test. Force Vuse, measure Iuse. If exceeds Io then DUT failed pre-test. Set Fmode=1, Vbd=Vuse and exit algorithm.

6) Ramp voltage from Vstart in equal Vstep increments for step time until breakdown detected or Vmax is reached.

   a) Measure current and step time at each step. Use to calculate Qbd.

   b) Monitor previous and successive I-V data points to detect breakdown using current compliance, 10X change in current, slope change and 1000x change.

7) Perform post test. Set Failure mode, breakdown detection method and test stop method.

8) Plot/save I-V data according to the value of Plot$.

9) Return Vbd, Ebd, Qbd, Fmode, Iuse, Ibd, Vcrit, Vbox.

Error Code Summary

This value is assigned to the output variable if an error occurred during algorithm execution:

Value	Meaning
1E-17	Initial Failure
1E+17	Oxide did not breakdown during ramp
2.0E+30	Input parameter error
2.1E+30	Step size that the user has requested is too small
1.5E+30	Analysis not performed
+XE+32	Operating system error "X" occurred during execution of algorithm

Compatibility with Previous Versions

XVRAMP_CAP, is provided to act as a parameter translation shell to maintain compatibility with HP BASIC test programs written for the obsolete **XVRAMP_CAP**. This algorithm, which calls **VRAMP_3_CAP,** is briefly described below.

XVRAMP_CAP

Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
V1†	Real	Volts	3.3	0.1~10	Start gate voltage
V2†	Real	Volts	19	1~200	Ending gate voltage
Vstep	Real	Volts	0.2	0.01~5	Gate bias step size
Tox	Real	Å	75	10~10E3	Design oxide thickness
Vuse†	Real	Volts	3.3	0.1~10	Post test check voltage
Icomp	Real	Amps	1.0E-1	1.0E-9~0.35	Compliance limit of sweep
Ithreshold	Real	Amps	1.0E-10	1.0E-15~1.0E-2	Lowest current level of search
Ramp_rate	Real	MV/cm•sec	1.0	0.01~15.0	Electric field ramp rate
Plot$	String	——	"Y"††		Plot/Save Data

† Must be specified as a positive number. The algorithm will determine the correct polarities based on the bulk type (N or P).

††A "Y" plots current versus voltage on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files*.

Device Variables*

Var Name	Var Type	Comment/Description
Area	Real	cm^2, Area of active region
Per†	Real	cm, Perimeter of active region
Tty$†	String	(N/P/Al), Top electrode type
Bty$	String	(N/P), Dopant type of bulk region

† Not used by WLR Oxide Algorithms

*These are the input variables for the CAP Device. A transistor structure may also be used provided the gate and bulk are correctly defined.

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Vbr	Real	Volts	Breakdown voltage
Fbr	Real	MV/cm	Breakdown field
Qbd	Real	C/cm^2	Charge to breakdown
Fmode	Real	——	Failure mode 1 - Failed initial test 2 - Catastrophic breakdown 3 - Masked catastrophic breakdown 4 - Non-catastrophic breakdown 5 - Other
Ileak	Real	$\log_{10}$ Amp	Current at V1

Algorithm Syntax

The syntax for calling the algorithm directly is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction to Oxide Integrity* section.

Xvramp_cap (V1, V2, Vstep, Tox, Vuse, Icomp, Ithreshold, Ramp_rate, Plot$,
Pins(*), Area, Per, Tty$, Bty$,
Vbr, Fbr, Qbd, Fmode, Ileak)

Methodology

The algorithm is a translation shell which calls the **VRAMP_3_CAP** algorithm to conduct the test. The shell maintains compatibility with existing HP BASIC test programs by changing/adding the following input/output parameters in the call from **XVRAMP_CAP** to **VRAMP_3_CAP**:

Input parameters:

V1: Renamed to Vstart.

V2: Renamed to Vmax.

Ithreshold: Renamed Icrit.

Io: Set to 1E-6 to be consistent with previous version.

Testcon: Added and set to 1 to always test in accumulation.

Output parameters:

Ileak: Returned as current during Vuse pre-test and not as V1 test.

Fmode: Returned with additional failure information.

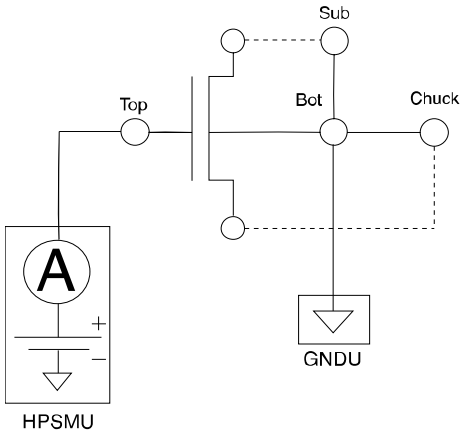
Ibd, Vcrit, Vbox: Not returned to **XVRAMP_CAP**.

VTDDB_3_CAP

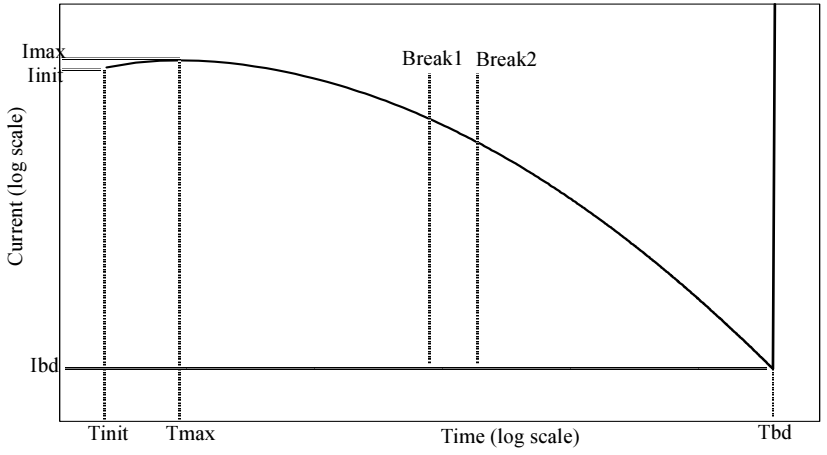
Performs constant voltage time dependent dielectric oxide breakdown in accumulation or inversion.

This routine measures the time-to-breakdown for a constant voltage stress. **VTDDB_3_CAP** provides a stress closer to operating conditions than **VRAMP_3_CAP** and can provide higher detail in breakdown data for modeling and prediction.

Test Configuration



Electrical Characteristics



Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Time_max	Real	Secs	300	1~1E6	Maximum time to wait for break-down
Estress†	Real	MV/cm	13	3~25	Field at which capacitor is stressed
Tox	Real	Å	75	10~10E3	Design oxide thickness
Vuse†	Real	Volts	3.3	0.1~20.0	Pre-/Post-test check voltage
Icomp	Real	Amps	0.01	0.001~0.35	Current compliance of SMU
Io	Real	Amps	1E-6	1E-9~1E-3	Pre-/Post-test current failure criteria
Testcon	Integer		1	1~2	Test in: 1=accumulation 2=inversion
Plot$	String	——	"Y"††		Plot/Save Data

† Certain combinations of these two variables may not be allowed due to SMU compliance considerations - the code will exit with a parameter error in this case. Enter as positive number. Code will determine polarity based on type (N or P) and test condition (accumulation or inversion).

††A "Y" plots current versus time on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files*.

Device Variables

Var Name	Var Type	Comment/Description
Area	Real	cm^2, Area of active region
Per†	Real	cm, Perimeter of active region
Tty$†	String	(N/P/Al), Top electrode type
Bty$	String	(N/P), Dopant type of bulk region

† Not used by WLR Oxide Algorithms

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Tbd	Real	Sec	Breakdown time
Vbd	Real	Volts	Stress voltage
Ibd	Real	Amps	Current at breakdown time, Tbd
Qbd	Real	C/cm^2	Accumulated charge
Iuse	Real	Log_{10} Amp	Leakage at Vuse in pre-test
Fmode	Real	——	Failure mode 1 - Failed initial test 2.X - Catastrophic breakdown 3.Y - Masked catastrophic breakdown 4.X - Non-catastrophic breakdown 5.Y - Other X=1,2 is breakdown method Y=depends on Qbd
Tinit	Real	Secs	First time step
Iinit	Real	Amps	Current at Tinit
Tmax	Real	Secs	Time at Imax
Imax	Real	Amps	Maximum measured current during stress

Algorithm Syntax

The syntax for calling the algorithm directly (that is, outside of Agilent SPECS) is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction to Oxide Integrity* section.

Vtddb_3_cap(Time_max, Estress, Tox, Vuse, Icomp, Io, Testcon, Plot$,
Pins(*), Area, Per, Tty$, Bty$,
Tbd, Vbd, Ibd, Iuse, Fmode, Tinit, Iinit, Tmax, Imax)

Theory of Operation

The *Introduction to Oxide Integrity* section provides a general overview.

VTDDB_3_CAP measures the time-to-breakdown of an insulator during a constant voltage stress. The advantage to using wafer level TDDB testing comes from its higher sensitivity to small changes in processing. Unlike the **VRAMP_3_CAP** routine, which has fixed step times (usually around 100ms) and hence fixed time resolution, the **VTDDB_3_CAP** routine operates as fast as permitted by the tester and algorithm. Another advantage is that a constant voltage stress is closer to operating conditions than the step stress performed in a voltage ramp test. This makes it easier to understand the **VTDDB_3_CAP** results.

Methodology

A constant voltage oxide stress is performed in the following manner. First, a pre-stress oxide test is performed to determine if the capacitor is leaky. Vuse is applied across the capacitor and the current Iuse is measured. If Iuse is greater than Io, the capacitor is considered to be an initial failure. If the Iuse is less than Io, then the stress voltage (Vstress) is applied and the current is a measured as a function of stress time. At each current measurement, the stress time is measured and used to calculated the integrated charge density to breakdown (Qbd). The time resolution depends on the test system measurement speed and CPU controller accuracy. Vstress is calculated based on the JESD35 recommendation as Vstress = Estress*Tox. The algorithm will stop when the current reaches the user-defined compliance (Icomp), 10x increase in current or Time_max is exceeded. A post-stress test (again by applying Vuse) is then performed to verify the oxide breakdown.

VTDDB_3_CAP also stores the current (Ibd) at the breakdown time (Tbd), and the leakage current (Iuse) at Vuse. The current-time curve is further parameterized by Tinit, Init, Imax and Tmax. Tinit is the stress time at the first time step. This value is useful for analyzing defect data. Iinit is the current at Init. Imax is the maximum current during the stress which occurs at the time Tmax.

The **VTDDB_3_CAP** algorithm determines the failure mode in a manner similar to JEDEC standard JESD35. The failure modes are classified into five categories according to how they respond to the initial pre-stress test, stress-test and post-stress test. These failure modes are defined in the table below.

Failure Mode (Fmode)	Initial Test	Stress Test	Post Test
1-Initial	Fail	N/A	N/A
2.X-Catastrophic	Pass	Fail	Fail
3.Y-Masked Catastrophic	Pass	Pass	Fail
4.X-Non Catastrophic	Pass	Fail	Pass
5.Y-Others	Pass	Pass	Pass

X is 1 or 2 depending on breakdown detection method. The breakdown detection methods are described in the following table.

Y=1 is normal exit, Y=2 means Qbd exceeded 50 C/cm^2

Breakdown Detection Method	Failure Mode 2 (Catastrophic)	Failure Mode 4 (Non-Catastrophic)
1 - Compliance	2.1	4.1
2 - 10x	2.2	4.2

Testing Notes: For fast wafer-level test conditions, the algorithm has a time step resolution of 3 ms to 10 ms. The first time step, Tinit, may take longer than this (in the range of 30 ms to 100 ms) on the Agilent 4062UX. For the Agilent 4070 series, Tinit is significantly less (in the range of 3ms to 10ms).

Because a significant amount of current-time data can be accumulated during a TDDB stress (e.g. once every 5 ms), the current-time data output by the Plot$ command is conditioned. The total output data is limited to 2048 data points. If the stress requires more than 2048 points, then the first 300 data points and the last 300 data points are stored. In between these points, the data is reduced so that only 100 points per decade of time are stored. Because of the real time sampling algorithm, there will be a small break between the first 300 data points and the first logarithmic sampling data point. This is shown in the electrical characteristic by the Break 1 and Break 2 lines. Because Qbd and other variables are calculated in real time using all the data, this break does not impact the accuracy of the output variables.

Data File Format: When using the Plot$ variable to output ASCII data to a file the resulting format will be:

Input parameters	
Output parameters	
Time (sec)	Current (A)
<data>	<data>

Test Structure

See "Test Structures" in the *Introduction to Oxide Integrity* section.

Step-by-Step Action

1) Check for legal input values. If any are out of range, output variables are set to flag values 2.0E+30 and algorithm exits.
2) Calculate stress voltage from stress field. Vstress = Estress*Tox.
3) Determine polarity based on substrate type and test condition (accumulation or inversion).
4) Perform pre-stress leaky oxide test. Apply Vuse, Measure current. If current exceed Io, set Fmode=1 and values to 1.0E-17, exit algorithm.
5) Apply stress voltage to capacitor and wait for breakdown to occur or time to exceed Time_max.
   a) Measure current as a function stress time. Use to calculate Qbd.
   b) If current reaches Icomp, then breakdown occurred.
6) Perform post leakage test. Set Failure mode.
8) Plot/save data according to value of Plot$.
9) Return Tbd, Vbd, Ibd, Iuse, Fmode, Tinit, Iinit, Tmax, Imax.

Error Code Summary

This value is assigned to the output variable if an error occurred during algorithm execution:

Value	Meaning
1E-17	Failed initial test at Vuse; leaky capacitor
1E+17	Test exceeded Time_max; no breakdown in the time allotted
2.0E+30	Input parameter error
1.5E+30	Analysis not performed
+XE+32	Operating system error "X" occurred during execution of algorithm.

Compatibility with Previous Versions

TDDBN_CAP, is provided to act as a parameter translation shell to maintain compatibility with HP BASIC test programs written for the obsolete **TDDBN_CAP**. This algorithm, which calls **VTDDB_3_CAP,** is briefly described below.

TDDBN_CAP

Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Time_max	Real	Secs	300	1~1E6	Maximum time to wait
Stress†	Real	MV/cm	13	3~25	Field at which capacitor is stressed
Tox	Real	Å	75	10~10E3	Design oxide thickness
Vuse	Real	Volts	3.3	0.1~10.0	Pre test check voltage
Comp†	Real	Amps	0.01	0.001~0.35	Current compliance of SMU
Plot$	String	——	"Y"††		Plot/Save Data

† Certain combinations of these two variables may not be allowed due to SMU compliance considerations - the code will exit with a parameter error in this case.

††A "Y" plots current versus time on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files*.

Device Variables

Var Name	Var Type	Comment/Description
Area	Real	cm^2, Area of active region
Per†	Real	cm, Perimeter of active region
Tty$†	String	(N/P/Al), Top electrode type
Bty$	String	(N/P), Dopant type of bulk region

† Not used by WLR Oxide Algorithms

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Tbr	Real	Sec	Breakdown time
Vbr	Real	Volts	Stress voltage
Fbr	Real	MV/cm	Stress field
Ileak	Real	Log_{10} Amp	Leakage at Vuse

Algorithm Syntax

The syntax for calling the algorithm directly is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction to Oxide Integrity* section.

Tddbn_cap(Time_max, Stress, Tox, Vuse, Comp, Plot$,
Pins(*), Area, Per, Tty$, Bty$,
Tbr, Vbr, Fbr, Ileak)

Methodology

The algorithm is a translation shell which calls the V**TDDB_3_CAP** algorithm to conduct the test. The shell maintains compatibility with existing HP BASIC test programs by changing/adding the following input/output parameters in the call from **TDDBN_CAP** to **VTDDB_3_CAP**:

Input parameters:

Io: Added and Set to 1E-6 to match hardcoded value.

Testcon: Added and set to 1 to always test in accumulation.

Stress: Renamed to Estress. Same meaning.

Comp: Renamed to Icomp. Same meaning.

Output parameters:

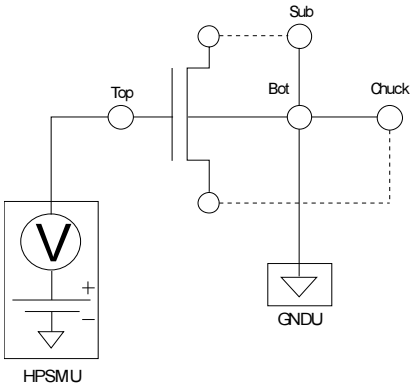
Tinit, Iinit, Tmax, Imax, Fmode: Not returned to **TDDBN_CAP**.

JRAMP_3_CAP

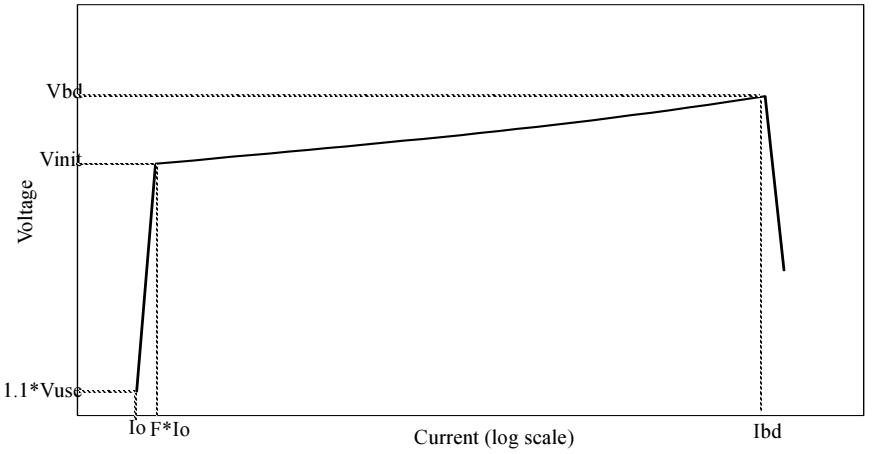
Oxide breakdown by current ramp in compliance with JEDEC Standard JESD35.

JRAMP_3_CAP current is increased in logarithmic steps until the oxide ruptures. The current and time step is measured at each interval and used to calculated accumulated charge per area (Qbd). Breakdown (or failure) is defined by a user-specified drop in voltage. Test can be done in accumulation or inversion (with proper structure) or on a MOSFET device. Optional connection(s) for minority carrier contact to permit breakdown in inversion is shown by dashed lines.

Test Configuration



Electrical Characteristics



Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Io†	Real	Amps	1E-6	1E-7~1E-3	Initial ramp current (pre/post check current)
Imax†	Real	Amps	0.001	1E-6~1E-1	Maximum ramp current
F	Real	——	1.25	1.0001~3.2	Current increase factor
Ramp_rate	Real	Decade/Sec	2	.1~5	Current ramp rate
Vuse†	Real	Volts	3.3	0.1~20	Pre and Post test compliance voltage
Vcomp	Real	Volts	100	1~200	Voltage compliance of current ramp
Fcrit	Real		0.9	0.5~0.999	Voltage failure criteria (%/100)
Testcon	Integer		1	1~2	Test in: 1=accumulation, 2=inversion
Plot$	String	——	"Y"††		Plot/Save Data

† Must be specified as a positive number. The algorithm will determine the correct polarities based on the bulk type (N or P) and test condition (accumulation or inversion).

††A "Y" plots voltage versus current on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files*.

Device Variables*

Var Name	Var Type	Comment/Description
Area	Real	cm^2, Area of active region
Per†	Real	cm, Perimeter of active region
Tty$†	String	(N/P/Al), Top electrode type
Bty$	String	(N/P), Dopant type of bulk region

† Not used by WLR Oxide Algorithms

*These are the input variables for the CAP Device. A properly designed transistor structure may also be used provided the gate, bulk pins are correctly defined. Note that a PMOS (NMOS) transistor structure has a N (P) Bty$.

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Ibd	Real	Amps	Breakdown current
Vbd	Real	Volts	Voltage at Ibd or max. voltage
Qbd	Real	C/cm^2	Accumulated charge
Fmode	Real	——	Failure mode 1.X - Failed initial test 2 - Catastrophic breakdown 3.Y - Masked catastrophic breakdown 4 - Non-catastrophic breakdown 5.Y - Other X=0,1 is initial test failure condition Y=1,2 is test stop condition
Vinit	Real	Volts	Voltage at F*I_o

Algorithm Syntax

The syntax for calling the algorithm directly (that is, outside of Agilent SPECS) is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction to Oxide Integrity* section.

Jramp_3_cap(Io, Imax, F, Ramp_rate, Vuse, Vcomp, Fcrit, Testcon, Plot$,
Pins(*), Area, Per, Tty$, Bty$,
Ibd, Vbd, Qbd, Fmode, Vinit)

Theory of Operation

The *Introduction to Oxide Integrity* section provides a general overview.

JRAMP_3_CAP measures the oxide breakdown current, voltage and accumulated charge by increasing the applied current until the oxide breaks down. Oxide breakdown has been explained by the generation of neutral electron traps as well as hole generation. Neutral electron traps are generated by the current flow through the insulator. If enough of these traps are generated to form a conductive path of traps between the gate and substrate, thermal breakdown occurs.

Methodology

The methodology for performing a current-ramp oxide breakdown is described in JEDEC standard JESD35. A pre-ramp voltage test is first performed. Vuse is applied to the capacitor. If the measured current is greater than Io, the capacitor is a Class A initial failure. If the capacitor has not failed, a pre-ramp current test is then performed. Io is applied to the capacitor. If the voltage across the capacitor fails to reach Vuse in a specified period of time (code default is 50ms), then the capacitor is a Class B initial failure. If the voltage reaches Vuse, then the current is ramped from Io*F in logarithmic current steps until breakdown is detected or Imax is reached. A post ramp test (again by applying Io) is performed to verify the oxide breakdown.

JRAMP_3_CAP measures the current just before oxide breakdown (Ibd) and the corresponding voltage (Vbd). An oxide breakdown is defined when the voltage drops by a user specified fraction (Fcrit) of the voltage at the previous step. The initial voltage (Vinit) at Io*F is also reported.

At each current step, the algorithm calculates the charge by multiplying the measured current by the measured step time. The charge at each step are added together to determine the charge to breakdown (Qbd).

The **JRAMP_3_CAP** algorithm determines the failure mode of each ramp test. The failure modes are classified according to initial pre-ramp tests, ramp test and post-ramp test. These failure modes are defined in the table below.

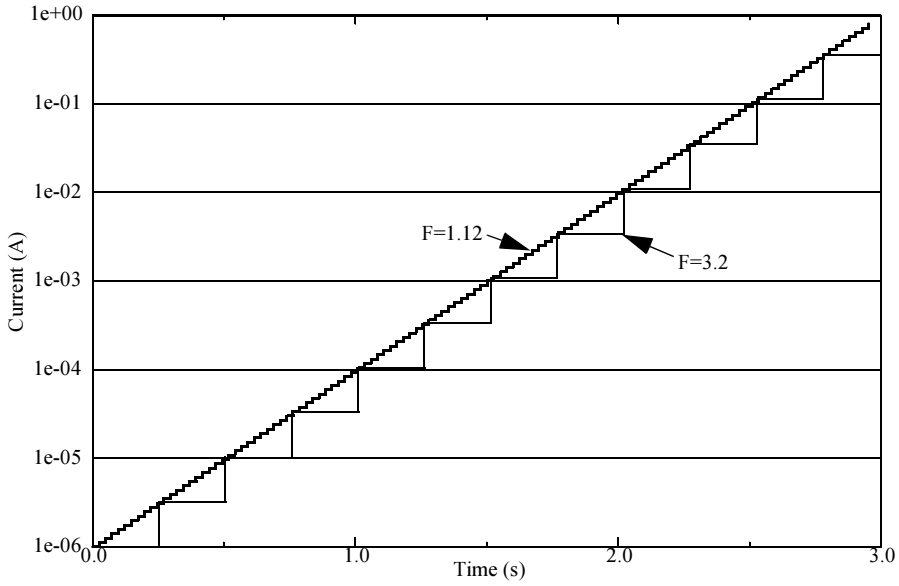
Failure Mode (Fmode)	Initial Voltage Test	Initial Test	Ramp Test	Post Test
1-Initial (Class A)	Fail	N/A	N/A	N/A
1.1-Initial (Class B)	Pass	Fail	N/A	N/A
2-Catastrophic	Pass	Pass	Fail	Fail
3.Y-Masked Catastrophic†	Pass	Pass	Pass	Fail
4-Non Catastrophic	Pass	Pass	Fail	Pass
5.Y-Others†	Pass	Pass	Pass	Pass

†Y is 1 or 2 depending on test stop condition.

Most test structures, if properly designed and tested, will fail catastrophically (Fmode=2). In other words, they will break down during the ramp and remain shorted between the gate and substrate (or well). Sometimes, however, the energy dissipated in the breakdown is enough to clear the short, leading to an unshorted gate and a non-catastrophic failure (Fmode=4). An Fmode of 4 could also occur if Vcomp is reached during the test. This could indicate an open circuit due to test fixture or test structure problems. The masked catastrophic (Fmode=3.Y) appears to pass the ramp tests, but then doesn't work post test. This type of failure will occur if the capacitor breaks down in a manner which is not detected during the ramp. The failure mode 5.Y indicates that the capacitor did not break down. The "Y" in these last two modes indicates how the test was stopped. If Y=1, then the current ramp finished at Imax. This may indicate a test fixture or test structure problem. If Y=2, then the test was stopped because Qbd exceeded 50 C/cm^2. JESD35 recommends a maximum Qbd (Qmax) of 50 C/cm^2.

Testing notes: The choice of current increase factor, F, is extremely important. Too large a value leads to a very coarse estimate of Qbd (because the last current step contributes the most to the value of Qbd); while a value too near 1.0 leads to a very slow experiment. F should also remain the same throughout testing in order to obtain comparable values for Qbd. An F greater than 1.5 is not recommended.

The graph below illustrates the effect F has on the step size using the JEDEC standard current ramp rate of 2 decades/sec. Note that a large F results in a large step size and the last current step contributing the most to the Qbd value.



Due to many possible combinations for F and ramp_rate, the step time may be smaller than a minimum allowable time. In this case the code will abort the test rather than continue. In order to avoid this possibility, the user can calculate the step time from the following:

$$\text{Step\_time} = \log10(F) / \text{Ramp\_rate} > \text{Min. time}$$

The minimum time (Min. time) in the algorithm is set to 10 ms. This is a conservative estimate for algorithm controlling a Agilent 4070 series test system. This 10 ms is a reasonable estimate the algorithm controlling a Agilent 4062UX. In any event, the actual step time is measured at each current step and is output with the I-V plot data.

JEDEC requirements for a current ramp specifies that the Ramp_rate be 2 decades per second with a step size of at most 50 ms.

Because of UNIX operating system scheduling, the user may occasionally observe a longer than expected step time for one data point during a ramp. This is a function of the UNIX operating system and will occur much less than 1% of the time. This does not effect the output parameters such as the Qbd since Qbd is calculated based on the measured step time and current.

Data File Format: When using the Plot$ variable to output ASCII data to a file the resulting format will be:

Input parameters		
Output parameters		
Step_time (secs)	Current (A)	Voltage (V)
<data>	<data>	<data>

Test Structure

See "Test Structures" in the *Introduction to Oxide Integrity* section.

Step-by-Step Action

1) Check for legal input values. If any are out of range, output variables are set to flag value 2.0E+30 and algorithm exits
2) Connect SMUs as in Test Configuration diagram.
3) Determine polarity based on substrate type and test condition (accumulation or inversion).
4) Calculate step time to produce user specified ramp_rate. If too small, set output values to 2.1E+30, exit algorithm.
5) Perform pre-ramp tests.
   a) Force Vuse, Measure I. If I > Io, then DUT failed Class A pre-test. Set Fmode=1 and exit algorithm.
   b) Force Io with voltage compliance of 1.1*Vuse, wait for 50ms. If voltage is below Vuse, then DUT failed Class B pre-test. Set Fmode=1.1 and exit algorithm.
6) Ramp current from Io*F in logarithmic steps for a fixed step time until breakdown is detected, Imax is reached or Vcomp reached.

a) Force stress current. Measure step time at each step and use to calculate Qbd.

b) Measure voltage. If voltage drops by Fcrit of voltage at previous step, then breakdown detected.

c) If measured voltage is within 1% of Vcomp, then end test.

7) Perform post ramp test. Set failure mode.

8) Plot/save I-V data according to value of Plot$.

9) Return Ibd, Vbd, Qbd, Fmode, Vinit.

Error Code Summary

This value is assigned to the output variable if an error occurred during algorithm execution:

Value	Meaning
1E-17	Initial Failure
1E+17	Oxide did not breakdown during ramp
1.5E+30	Analysis not performed
2.0E+30	Input parameter error
2.1E+30	Step time is too short
+XE+32	Operating system error "X" occurred during execution of algorithm

Compatibility with Previous Versions

JRAMP_CAP, is provided to act as a parameter translation shell to maintain compatibility with HP BASIC test programs written for the obsolete **JRAMP_CAP**. This algorithm, which calls **JRAMP_3_CAP,** is briefly described below.

JRAMP_CAP

Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
I_o†	Real	Amps	1E-6	1E-7~1E-3	Initial current
I_final†	Real	Amps	0.001	1E-6~1E-1	Final current
F	Real	——	1.25	1.0001~3.2	Current increase factor
Ramp_rate	Real	Decade/Sec	2	.1~5	Current ramp rate
Vuse†	Real	Volts	3.3	0.1~10	Pre and Post test voltage
Plot$	String	——	"Y"††		Plot/Save Data

† Must be specified as a positive number. The algorithm will determine the correct polarities based on the bulk type (N or P).

††A "Y" plots voltage versus current on screen. See section PDQ-WLR Algorithm Inputs for Graphics and CSV Files.

Device Variables

Var Name	Var Type	Comment/Description
Area	Real	cm^2, Area of active region
Per†	Real	cm, Perimeter of active region
Tty$†	String	(N/P/Al), Top electrode type
Bty$	String	(N/P), Dopant type of bulk region

† Not used by WLR Oxide Algorithms

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Ibr	Real	Amps	Breakdown current
Vbr	Real	Volts	Voltage required to rupture oxide
Qbd	Real	C/cm^2	Accumulated charge
Fmode	Real	——	Failure mode 1 - Failed initial test 2 - Catastrophic breakdown 3 - Masked catastrophic breakdown 4 - Non-catastrophic breakdown 5 - Other

Algorithm Syntax

The syntax for calling the algorithm directly is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction to Oxide Integrity* section.

Jramp_cap(I_o, I_final, F, Ramp_rate, Vuse, Plot$,
Pins(*), Area, Per, Tty$, Bty$,
Ibr, Vbr, Qbd, Fmode)

Methodology

The algorithm is a translation shell which calls the **JRAMP_3_CAP** algorithm to conduct the test. The shell maintains compatibility with existing HP BASIC test programs by changing/adding the following input/output parameters in the call from **JRAMP_CAP** to **JRAMP_3_CAP**:

Input parameters:

I_o: Renamed to Io. Same meaning.

I_final: Renamed to Imax. Same meaning.

Vcomp: Added and set to 100V for consistency.

Fcrit: Added and set to 0.85 for consistency.

Testcon: Added and set to 1 to always test in accumulation.

Output parameters:

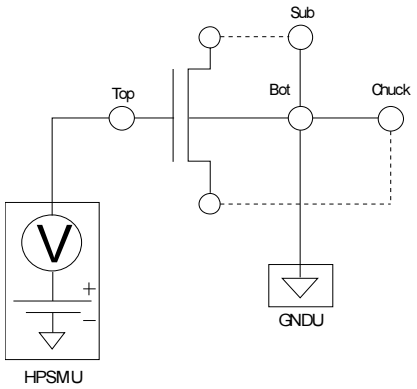
Vinit: Not returned to **JRAMP_CAP**.

JTDDB_3_CAP

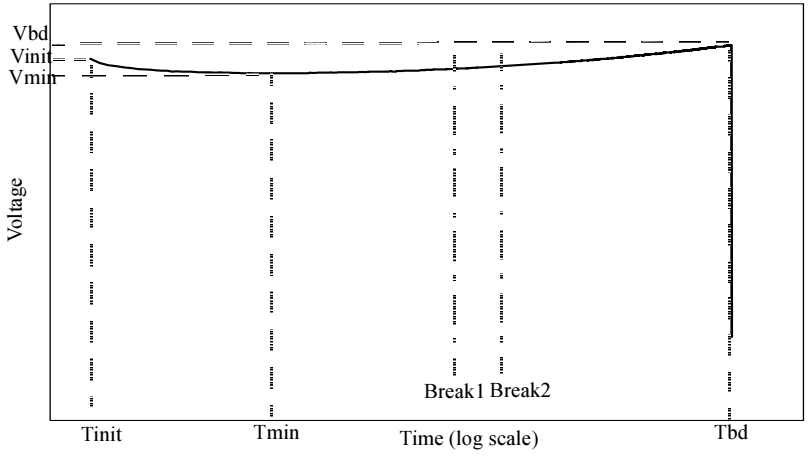
Performs constant current time dependent dielectric oxide breakdown in accumulation or inversion.

This routine measures the time-to-breakdown for a constant current stress. Testing can be done in accumulation or inversion (with proper test structure) as well as with MOSFET structures. **JTDDB_3_CAP** is the constant current analog of the current ramp algorithm **JRAMP_3_CAP**.

Test Configuration



Electrical Characteristics



Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Time_max	Real	Secs	300	1~1E6	Maximum time to wait for break-down
Jstress†	Real	A/cm^2	1	1E-6~20	Stressing current density
Vcomp†	Real	Volts	24	1~200	Voltage compliance for current stress
Vuse†	Real	Volts	3.3	0.1~20.0	Pre/Post test voltage
Io	Real	Amps	1E-6	1E-9~1E-3	Pre/Post test current failure criteria
Fcrit	Real		0.9	0.5~0.999	Voltage failure criteria (%/100)
Testcon	Integer		1	1~2	Test in: 1=accumulation 2=inversion
Plot$	String	——	"Y"††		Plot/Save Data

†Certain combinations of these two variables may not be allowed due to SMU compliance considerations - the code will exit with a parameter error in this case. Put in as positive number. Code will determine polarity based on type (N or P) and test condition (accumulation or inversion).

††A "Y" plots voltage versus time on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files*.

Device Variables

Var Name	Var Type	Comment/Description
Area	Real	cm^2, Area of active region
Per†	Real	cm, Perimeter of active region
Tty$†	String	(N/P/Al), Top electrode type
Bty$	String	(N/P), Dopant type of bulk region

† Not used by WLR Oxide Algorithms

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Tbd	Real	Sec	Time for breakdown
Vbd	Real	Volts	Voltage at breakdown
Ibd	Real	Amps	Stress Current
Qbd	Real	C/cm^2	Accumulated charge
Iuse	Real	Log_{10} Amp	Leakage at Vuse during Pre-test
Fmode	Real		Failure mode 1 - Failed initial test 2 - Catastrophic breakdown 3 - Masked catastrophic breakdown 4 - Non-catastrophic breakdown 5 - Other
Tinit	Real	Secs	First time step
Vinit	Real	Volts	Voltage at Tinit
Tmin	Real	Secs	Time at Vmin
Vmin	Real	Secs	Minimum voltage during stress

Algorithm Syntax

The syntax for calling the algorithm directly (that is, outside of Agilent SPECS) is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction to Oxide Integrity* section.

Jtddb_3_cap(Time_max, Jstress, Vcomp, Vuse, Io, Fcrit, Testcon, Plot$,
Pins(*), Area, Per, Tty$, Bty$,
Tbd, Vbd, Ibd, Qbd, Iuse, Fmode, Tinit, Vinit, Tmin, Vmin)

Theory of Operation

The *Introduction to Oxide Integrity* section provides a general overview.

JTDDB_3_CAP measures the time-to-breakdown of an insulator during a constant current density stress. The advantage to using wafer level TDDB testing comes from its higher sensitivity to small changes in processing. Unlike the **JRAMP_3_CAP** routine, which has fixed step times (usually around 50ms) and hence fixed time resolution, the **JTDDB_3_CAP** routine operates as fast as permitted by the tester and algorithm. As a result, an artificial binning of the data does not take place in **JTDDB_3_CAP**. The only binning limit for this code is the test time resolution.

Methodology

The constant current oxide stress is performed in the following manner. First, a pre-stress oxide test is performed to determine if the capacitor is leaky. Vuse is applied across the capacitor and the current (Iuse) is measured. If Iuse is greater than Io, the capacitor is considered to be an initial failure. If the Iuse is less than Io, then the stress current (Istress) is applied and the voltage is measured as a function of stress time. At each voltage measurement, the stress time is measured. The time resolution depends on the test system measurement speed and CPU controller. Istress is calculated as Istress = Jstress*Area. The algorithm will stop when the voltage drops by user-defined fraction (Fcrit) from the previous measured value or Time_max is exceeded. A post-stress test (again by applying Vuse) is then performed to verify the oxide breakdown.

JTDDB_3_CAP stores the voltage (Vbd) at breakdown time (Tbd), and the leakage current (Iuse) at Vuse. The stress current (Ibd) is also output and the charge density to breakdown (Qbd) is simply Ibd*Tbd. The voltage-time curve is further parameterized by Tinit, Vinit, Vmin and Tmin. Tinit is the stress time at the first time step. This value is useful for analyzing defect data. Vinit is the voltage at Tinit. Vmax is the maximum voltage during the stress which occurs at the time Tmax.

The **JTDDB_3_CAP** algorithm determines the failure mode in a manner similar to JEDEC standard JESD35. The failure modes are classified into five categories according to how they respond to the initial pre-stress test, stress-test and post-stress test. These failure modes are defined in the table below.

Failure Mode (Fmode)	Initial Test	Stress Test	Post Test
1-Initial	Fail	N/A	N/A
2-Catastrophic	Pass	Fail	Fail
3-Masked Catastrophic	Pass	Pass	Fail
4-Non Catastrophic	Pass	Fail	Pass
5-Others	Pass	Pass	Pass

Plasma Damage

JTDDB_3_CAP can also be used along with the **VT_MOS** and/or **CPV_MOS** to monitor the susceptibility of an oxide to process-induced plasma damage. Special plasma damage structure(s) are first measured with **VT_MOS** and/or **CPV_MOS** algorithms. Then, a non-destructive **JTDDB_3_CAP** test is performed. A second **VT_MOS** and/or **CPV_MOS** is measured. The change in threshold voltage, transconductance or interface state density is used as an indicator of plasma damage and compared to a reference test structure exposed to the same stress. To quantitatively determine the plasma damage impact on device reliability, a short hot carrier stress (**HCI_3_MOS** or **HCSTRESS_MOS**) is performed on the special plasma damage structures. Additional information about plasma damage may be found in the *Plasma Damage* section of the *Hot Carriers and Device Reliability* Chapter.

Testing Notes: For fast wafer-level test conditions, the algorithm has a time step resolution of 3 ms to 10 ms. The first time step, Tinit, may take longer than this (in the range of 30 ms to 100 ms) on the Agilent 4062UX. For the Agilent 4070 series, Tinit is significantly less (in the range of 3ms to 10ms).

Because a significant amount of voltage-time data can be accumulated during a TDDB stress (e.g. once every 10ms), the voltage-time data output by the Plot$ command is conditioned. The total output data is limited to 2048 data points. If the stress requires more than 2048 points, then the first 300 data points and the last 300 data points are stored. In between these points, the data is reduced so that only 100 points per decade of time are stored. Because of the real time sampling algorithm, there will be a small break between the first 300 data points and the first logarithmic sampling data point. This is shown in the electrical characteristic by the Break 1 and Break 2 lines. Since the Qbd and other variables are calculated in real time using all the data, this break does not impact the accuracy of the output variables.

Data File Format: When using the Plot$ variable to output ASCII data to a file the resulting format will be:

Input parameters	
Output parameters	
Time (sec)	Voltage (V)
<data>	<data>

Test Structure

See "Test Structures" in the *Introduction to Oxide Integrity* section.

Step-by-Step Action

1) Check for legal input values. If any are out of range, output variables are set to flag values 2.0E+30 and algorithm exits.

2) Calculate stress current as Jstress*Area.

3) Determine polarity based on substrate type and test condition (accumulation or inversion).

4) Perform pre-stress leaky oxide test. Apply Vuse, measure current. If current exceeds Io, set Fmode=1 and values to 1.0E-17, exit algorithm.

5) Apply stress current to capacitor and wait for breakdown or time to exceed Time_max.

   a) Measure voltage as a function stress time.

   b) If measured voltage is Fcrit below the previous measured voltage, then breakdown detected.

   c) If measured voltage is Vcomp, then end stress.

6) Perform post leakage test. Set Failure mode.

7) Plot/save data according to value of Plot$.

8) Return Tbd, Vbd, Ibd, Qbd, Iuse, Fmode, Tinit, Vinit, Tmin, Vmin.

Error Code Summary

This value is assigned to the output variable if an error occurred during algorithm execution:

Value	Meaning
1E-17	Failed initial test at Vuse; leaky capacitor
1E+17	Test exceeded Time_max; no breakdown in the time allotted
1.5E+30	Analysis not performed
2.0E+30	Input parameter error
+XE+32	Operating system error "X" occurred during execution of algorithm

Compatibility with Previous Versions

JTDDB_3_CAP, is provided to act as a parameter translation shell to maintain compatibility with HP BASIC test programs written for the obsolete **TDDBI_CAP**. This algorithm, which calls **JTDDB_3_CAP,** is briefly described below.

TDDBI_CAP

Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Time_max	Real	Secs	300	1~1E6	Maximum time to wait
Jstress†	Real	A/cm^2	1	1E-6~10	Stressing current density
Vmax†	Real	Volts	24	1~200	Max. voltage to be applied to device
Vuse†	Real	Volts	3.3	0.1~10.0	Pre test check voltage
Plot$	String	——	"Y"††		Plot/Save Data

†Certain combinations of these two variables may not be allowed due to SMU compliance considerations - the code will exit with a parameter error in this case. Put in as positive number. Code will determine polarity based on type (N or P).

††A "Y" plots voltage versus time on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files.*

Device Variables

Var Name	Var Type	Comment/Description
Area	Real	cm^2, Area of active region
Per†	Real	cm, Perimeter of active region
Tty$†	String	(N/P/Al), Top electrode type
Bty$	String	(N/P), Dopant type of bulk region

† Not used by WLR Oxide Algorithms

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Tbr	Real	Sec	Breakdown time
Ibr	Real	Amps	Breakdown current
Qbd	Real	C/cm^2	Accumulated charge
Ileak	Real	Log_{10} Amp	Leakage at Vuse

Algorithm Syntax

The syntax for calling the algorithm directly is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction to Oxide Integrity* section.

Tddbi_cap (Time_max, Jstress, Vmax, Vuse, Plot$,
Pins(*), Area, Per, Tty$, Bty$,
Tbr, Ibr, Qbd, Ileak)

Methodology

The algorithm is a translation shell which calls the **JTDDB_3_CAP** algorithm to conduct the test. The shell maintains compatibility with existing HP BASIC test programs by changing/adding the following input/output parameters in the call from **TDDBI_CAP** to **JTDDB_3_CAP**:

Input parameters:

Io: Added and set to 1E-6 for consistency.

Fcrit: Added and set to 0.85 for consistency.

Vcomp: Same meaning as Vmax as previous code.

Testcon: Added and set to 1 to always test in accumulation.

Output parameters:

Fmode, Tinit, Vinit, Tmax, Vmax: New variables not returned.

Note: Vmax is an input in **TDDBI_CAP** and is now an output in **JTDDB_3_CAP**.

HFCV_CORR_3_CAP

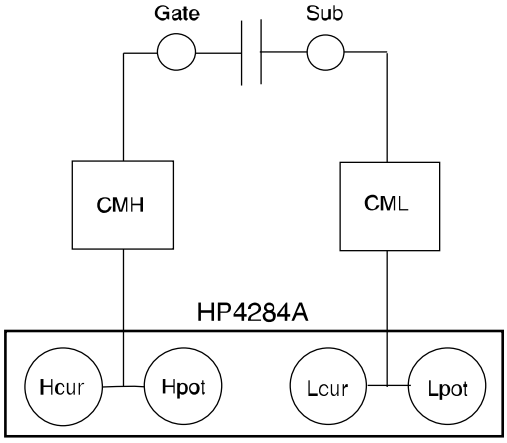
Pre high-frequency CV sweep meter correction for Agilent 4284A Meter.

HFCV_CORR_3_CAP incorporates the open and short correction and cable length selection methods discussed in the Agilent 4284A Precision LCR Meter Operation Manual, Agilent Part No. 04284-90010.

These procedures correct for stray admittance, residual impedance, and cable length measurement errors present when performing high frequency CV measurements using the test algorithms **HFCV_3_CAP** or **CV_3_CAP**.

Test Configuration

CMH and CML as labeled on the parametric tester.



Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
C_corr	Integer	Meters	0	0~4	Cable length correction
S_corr$	String	——	"N"	"N","Y"	Short correction (Y or N)
O_corr$	String	——	"N"	"N","Y"	Open correction (Y or N)
Sig_level	Real	Vrms	0.015	0.01~10	Test signal level

Device Variables

Var Name	Var Type	Comment/Description
Area	Real	cm^2, Area of active region
Per†	Real	cm, Perimeter of active region
Tty$†	String	(N/P/Al), Top electrode type
Bty$	String	(N/P), Dopant type of bulk region

† Not used by WLR Oxide Algorithms

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
None	——	——	——

Algorithm Syntax

The syntax for calling the algorithm directly (that is, outside of Agilent SPECS) is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction to Oxide Integrity* section.

Hfcv_corr_3_cap (C_corr, S_corr$, O_corr$, Sig_level, Pins(*))

Theory of Operation

This algorithm performs open and/or short correction of the Agilent 4284A LCR meter and test fixture configured as they would be during a high frequency CV (see the **HFCV_3_CAP** algorithm). This algorithm should be used each time a new probe card is installed or the test fixture configuration is changed. This will increase the accuracy of the **HFCV_3_CAP** algorithm test results, especially at high frequencies, by correcting for meter to test fixture cable length, stray admittance, and residual impedance.

If a wafer prober is being used, position the empty probe chuck under the probe tips to provide a surface to short the probe tips together. If a packaged part is being tested and no prober is available, the operator will be prompted to remove the device under test and manually open and short the test contacts. The algorithm should be set up and run as if a device were present so that the proper switching matrix contacts can be closed during the test.

Cable length selection is discussed in the Agilent 4284A Precision LCR Meter Operation Manual, Agilent Part No. 04284-90010. Open and short Correction are also discussed in the Operation Manual.

After running the **HFCV_CORR _3_CAP** algorithm, set the cable length, open, and short correction parameters in the **HFCV_3_CAP** algorithm.

Test Structure

None.

Step-by-Step Action

1) Check for legal input values. If any illegal values are present, the algorithm exits.
2) Check for legal hardware configuration. If hardware is missing or improperly configured an error message is displayed and the algorithm exits.
3) Connect Agilent 4284A meter as in Test Configuration diagram.
4) Initialize and set up Agilent 4284A meter (set range, corrections, signal level, frequency, etc.)

5) Prompt operator to position empty chuck under probe tips if prober is used, or remove device under test from test fixture if package part is used.

6) Raise chuck if prober is in use, or prompt operator to short gate and substrate connections together if package part.

7) Perform 90 second Short Correction.

8) Lower chuck if prober is in use, or prompt operator to remove short from test fixture so gate to substrate connections are open.

9) Perform 90 second Open Correction.

10) Disconnect hardware and exit algorithm.

Error Code Summary

This value is assigned to the output variable if an error occurred during algorithm execution:

Value	Meaning
2.0E+30	Input parameter error or meter error
1.5E+30	Analysis not performed
+XE+32	Operating system error "X" occurred during execution of algorithm.

Special Setup Instructions for Agilent 4284A

1) Connect the Agilent 4284 Hcur and Hpot connections together through a BNC "T" to the CMH port of the Agilent 4070 series test head or the Agilent 4085 switching matrix.

2) Connect the Agilent 4284A Lcur and Lpot connections together through a BNC "T" to the CML port of the Agilent 4070 series test head or the Agilent 4085 switching matrix.

Compatibility with Previous Versions

An algorithm, **HFCV_CORR_CAP**, is provided to act as a parameter translation shell to maintain compatibility with HP BASIC test programs written for the obsolete **HFCV_CORR_CAP**. This algorithm, which calls **HFCV_CORR_3_CAP,** is briefly described below. It is highly recommended that test plans be modified to use **HFCV_CORR_3_CAP** because of enhanced functionality.

HFCV_CORR_CAP

Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
C_corr	Integer	Meters	0	0~4	Cable length correction
S_cor$	String	——	"N"	"N","Y"	Short correction (Y or N)
O_cor$	String	——	"N"	"N","Y"	Open correction (Y or N)

Device Variables

Var Name	Var Type	Comment/Description
Area	Real	cm^2, Area of active region
Per†	Real	cm, Perimeter of active region
Tty$†	String	(N/P/Al), Top electrode type
Bty$	String	(N/P), Dopant type of bulk region

† Not used by WLR Oxide Algorithms

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
None	——	——	——

Algorithm Syntax

The syntax for calling the algorithm directly is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction to Oxide Integrity* section.

Hfcv_corr_cap(C_corr, S_corr$, O_corr$, Pins(*), Area, Per, Tty$, Bty$)

Methodology

The algorithm is a translation shell which calls the **HFCV_CORR_3_CAP** algorithm to conduct the test. The shell maintains compatibility with existing HP BASIC test programs by removing/adding the following input parameters in the call from **HFCV_CORR_CAP** to **HFCV_CORR_3_CAP**:

Input parameters:

Sig_level: Set to 0.015, not supported in **HFCV_CORR_CAP.**

Output parameters:

No changes.

HFCV_3_CAP

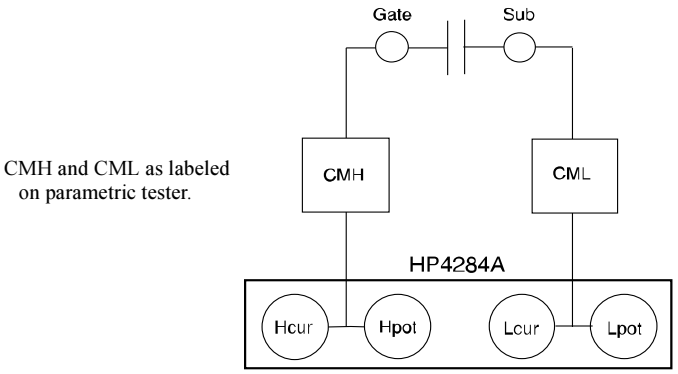
Measure high frequency CV curve of a MOS capacitor and extract physical parameters.

The **HFCV_3_CAP** algorithm sweeps a MOS capacitor and measures the capacitance. The flatband, midgap, threshold voltages, and substrate doping are extracted.

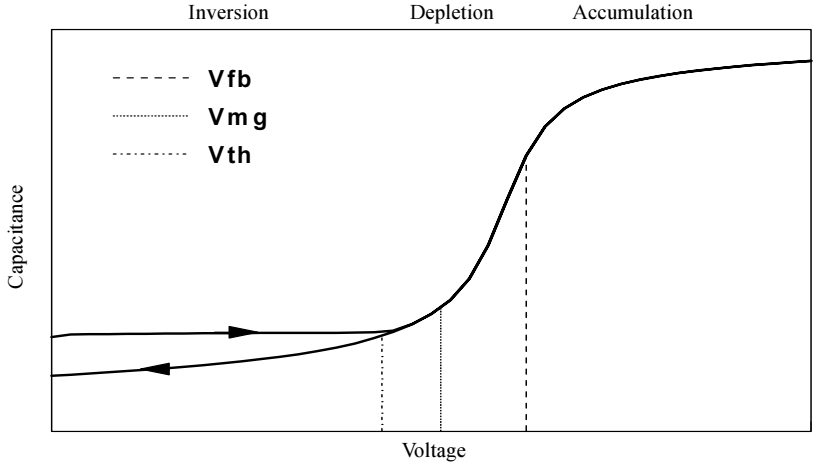
This test configuration requires an Agilent 4284A precision LCR Meter with the power amp option 001 installed. The 2m/4m cable length option 006 is recommended.

See *System Configuration* in the *Installation* chapter.

Test Configuration



Electrical Characteristics: High Frequency CV - N Substrate



Input Variables

Var Name[1]	Var Name 2	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Chktmp	A	Real	°C	27		Chuck temperature
C_corr	B	Integer	Meters	0	0~4	Cable length correction
S_cor$	C$	String	——	"N"	"N","Y"	Short correction (Y or N)
O_cor$	D$	String	——	"N"	"N","Y"	Open correction (Y or N)
Start†	E	Real	Volts	3.3	0~40	Start sweep voltage
Stop†	F	Real	Volts	3.3	0~40	Stop sweep voltage
Step†	G	Real	Volts	0.1	0.005~40	Sweep step size
Tdelay	H	Real	Seconds	0.05	0.01~60	Delay between each step
Freq	I	Real	Hz	1E6	20~1E+6	Test frequency
Level	J	Real	Vrms	0.025	0.01~10	Test signal level
Range	K	Real	Ω	0	0~1E5	Meter range
Integ	L	Integer	——	2	1,2,3	Meter integration time 1: Short, 2: Medium, 3: Long
Dirswp	M	Integer	——	3	1,2,3	Sweep direction 1: Inversion to accumulation 2: Accumulation to inversion 3: Both directions starting at inversion
Calc$	N$	String	——	"Y"	"N"~"Y"	Run analysis calculations?
Plot$	O$	String	——	"Y"††		Plot/Save Data

[1]Variable name under Agilent-ICMS and HP BASIC.

[2]Variable Name under Agilent SPECS.

† Must be specified as a positive number. The algorithm will determine the correct polarities based on the substrate type (N or P).

††A "Y" plots capacitance versus voltage on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files*.

NOTE: A prober light, serving as a source of minority carriers, is required when Calc$="Y" and Dirswp=1 or Dirswp=3. If the /opt/WLR/etc/algFlags "Prober Light" is enabled, then Plight_on_xl and Plight_off_xl will be called; otherwise, the system will prompt the user to provide an external light source. See section *Prober Interfaces for CV Users* in Chapter 2.

Device Variables

Var Name	Var Type	Comment/Description
Area	Real	cm^2, Area of active region
Per†	Real	cm, Perimeter of active region
Tty$†	String	(N/P/Al), Top electrode type
Bty$	String	(N/P), Dopant type of bulk region

† Not used by WLR Oxide Algorithms

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Cox	Real	Farads	Maximum capacitance in accumulation
Cmin	Real	Farads	Minimum capacitance in true inversion
Nb	Real	$atoms/cm^3$	Doping concentration
Vfb	Real	Volts	Flatband voltage
Vmg	Real	Volts	Midgap voltage
Vth	Real	Volts	Threshold voltage
Tox	Real	Angstroms	Calculated oxide thickness

Algorithm Syntax

The syntax for calling the algorithm directly (that is, outside of Agilent SPECS) is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction to Oxide Integrity* section.

Hfcv_3_cap(Chktmp, C_corr, S_cor$, O_cor$, Start, Stop, Step, Tdelay,
Freq, Level, Range, Integ, Dirswp, Calc$, Plot$,
Pins(*), Area, Per, Tty$, Bty$,
Cox, Cmin, Nb, Vfb, Vmg, Vth, Tox)

Theory of Operation

If this algorithm is to be used as a part of an oxide stress experiment, then The *Introduction to Oxide Integrity* section provides a general overview.

This algorithm performs a high frequency capacitance versus voltage (CV) measurement on a MOS capacitor. If the user desires, the routine can analyze the CV data and determine substrate doping (Nb), flatband voltage (Vfb), midgap voltage (Vmg) and threshold voltage (Vth). Determination of these parameters is described later. The reader unfamiliar with CV analysis techniques should review several works [1,2].

This routine is useful for characterizing process-induced oxide damage as well as performing routine process monitoring. Shifts in Vfb, Vmg or Vth can be used as a measure of charge trapped in the oxide. The routine **CV_3_CAP** uses the **HFCV_3_CAP** and the quasi-static CV routine (**QCV_3_CAP**) core subroutines to determine interface state density Dit at the Si-SiO_2 interface.

Substrate Doping

The substrate doping (Nb) is calculated based on the maximum-minimum capacitance technique [1]. This technique uses the maximum capacitance (Cmax) in accumulation and the true minimum capacitance in inversion (Cmin) to calculate Nb. It is important to sweep from inversion to accumulation, with a source of minority carriers provided at the start of the sweep, to obtain an accurate value of Cmin. Sweeping from accumulation will drive the capacitor into deep depletion, underestimating Cmin.

Treating the MOS system as two capacitors in series, the maximum depletion width in inversion, w_{max} is

$$W_{max} = \left(\varepsilon_{si} \cdot Area \cdot \left[\frac{1}{C_{min}} - \frac{1}{C_{max}}\right]\right) \qquad (1)$$

where ε_{Si} is permittivity of silicon and Area is the gate oxide area.

Using the depletion approximation for a uniform doping density, w_{max} may also be written as

$$W_{max} = \sqrt{\frac{2(\varepsilon_{si})\phi_{s,inv}}{q \cdot N_b}} \qquad (2)$$

where $\phi_{s,inv}$ is the surface potential in inversion.

$\phi_{s,inv}$ is approximated as

$$\phi_{s,inv} = 2.1 \cdot \phi_f + 1.33 \cdot kT \qquad (3)$$

where k = Boltzmann's constant; T = temperature; and ϕ_f = Fermi potential which is

$$\phi_f = k \cdot T \cdot \ln\left[\frac{N_b}{n_i}\right] \qquad (4)$$

where n_i is the intrinsic carrier concentration.

Equating (1) and (2), substituting (3) and (4), and simplifying, results in

$$\frac{N_b}{\left(2.1 \cdot \ln\left[\frac{N_b}{n_i}\right] + 1.33\right)} = \frac{2 \cdot k \cdot T \cdot \overline{C_{max}}^2}{q \cdot Area^2 \cdot \varepsilon_{si}} \cdot \left[\frac{C_{max}}{C_{min}} - 1\right]^{-2} \qquad (5)$$

Once a CV sweep has been done, all the parameters except N_b are known in (5). Since (5) is transcendental, it is solved by iteration for N_b.

Flatband, Midgap and Threshold Voltage

Flatband is the condition in a MOS system where the energy bands are flat. That is, the surface potential, ϕ_s is zero. ϕ_s is defined as the difference between the intrinsic energy and the Fermi level at the Si-SiO_2 interface. Midgap is the condition in a MOS system where ϕ_s equals ϕ_f (Fermi potential). We define threshold in a MOS system to occur when ϕ_s equals $2\phi_f$. In this case, the MOS system is heavily inverted at the Si-SiO_2 interface.

In order to determine these voltages, a theoretical capacitance curve is needed. If we treat the MOS system as two capacitors in series, then the total capacitance, $C(\phi_s)$, as a function of the surface potential is

$$\frac{1}{C(\phi_s)} = \frac{1}{C_{max}} + \frac{1}{C_s(\phi_s)} \qquad (6)$$

where $C_s(\phi_s)$ is the capacitance due to the semiconductor depletion width and C_{max} is the capacitance due to the oxide thickness. $C_s(\phi_s)$ can be solved numerically. However, a closed form approximation can be derived for a p-type substrate capacitor as [2]

$$C_s(V_s) = \frac{SGN(V_s)}{\sqrt{2}} \cdot C_{FBS} \cdot \frac{[1 - \exp((-V)_s)]}{\sqrt{(V_s - 1) + \exp((-V)_s)}} \qquad (7)$$

where $V_s = \phi_s / k \cdot T$ = normalized surface potential,

$C_{FBS} = \varepsilon_{si} / L_D$,

$$L_D = \text{Debye length} = \sqrt{\frac{\varepsilon_{si} \cdot k \cdot T}{q[N_b + n_i^2/(N_b)]}}$$, and

SGN = 1 if $V_s > 0$ and -1 if $V_s < 0$.

Equation 7 holds for all normalized potentials, V_s, less than a matching potential, V_m. For $V_s > V_m$, C_s is fixed at $V_s = V_m$. The matching potential is chosen to minimize error and is given by the Lindner potential [3], $V_m = 2.1U_f + 1.33$, where U_f is the normalized Fermi potential $U_f = \phi_s / kT$.

From (7), it can be seen that once one knows N_b and C_{max}, $C(\phi_s)$ is completely determined. Using (7), we calculate the capacitance at various surface potentials. For example the capacitance at flatband, C_{fb} is $C(\phi_s = 0)$, which is

$$C_{fb} = \frac{C_{fbs} \cdot C_{max}}{(C_{fbs} + C_{max})} \qquad .(8)$$

With this calculated value of C_{fb}, we then use the measured CV data to determine the gate voltage corresponding to C_{fb}. This is defined as the flatband voltage, V_{fb}. Similarly, we use (7) to determine the capacitance at midgap ($\phi_s = \phi_f$). Then using the measured CV curve, we obtain V_{mg}. Finally, we obtain V_{th} in the same manner.

An expression similar to (7) is used for n-type substrate capacitors [1].

Methodology

This algorithm sweeps the gate voltage from inversion to accumulation. The capacitance at each voltage step is measured after a user-defined delay time (Tdelay). In order to extract the correct physical parameters, the minimum capacitance in inversion, Cmin must be accurately determined. This is done by providing a source of minority carriers at the beginning of the sweep. In order to generate the needed minority carriers, the capacitor is exposed to a light source (either the prober microscope light or a manually operated light source). This is done for a few seconds after the device has been biased into depletion. The photons form enough electron-hole pairs to act as an almost instant source of minority carriers. The light source is removed and Cmin is then measured in a loop until it settles.

Caveats and Requirements

The CV measurements must be performed with the DUT shielded from light. This reduces the error due to photon-induced minority carrier generation.

The Agilent 4284A Option 001 (Power amp/DC bias) must be installed in the meter. This option is discussed in the Operations Manual [4].

The Agilent 4284A Option 006 (2m/4m Cable Length Operation) is highly recommended. This will allow for the use of cable correction and cable lengths greater than 1m (e.g. the Agilent 16048D 2m test leads or the Agilent 16048E 4m test leads). If this option is installed, you may select either 0, 1, 2 or 4 for the C_corr parameter in the **HFCV_3_CAP** algorithm. This option is discussed in the Operation Manual [4].

It is also highly recommended that the Agilent 4284A open and short correction be utilized. This is particularly important each time a probe card or test fixture is changed and the measurement frequency is greater than 100 kHz. The open correction cancels errors due to stray admittance in parallel with the capacitor being measured. The short correction corrects for residual impedance in series with the capacitor. These correction capabilities are easily set using the algorithm **HFCV_CORR_3_CAP**. After the **HFCV_CORR_3_CAP** algorithm has be performed, the O_corr and/or S_corr parameters in the **HFCV_3_CAP** algorithm may be set to "Y". These correction capabilities are discussed in the Agilent 4284A Operation Manual [4].

The measurement range can usually be set to automatic (Range = 0 in this algorithm). Nevertheless, Range may be set to one of nine specific values as discussed in the Agilent 4284A Operation Manual [4]. The capacitance range can be specified using this parameter, but the meter performs the impedance measurement according to the impedance range, as shown in the **Measure_cmu84** section of [5].The basic relationship between the capacitance and impedance range is defined by the following equation:

$$impedance = \frac{1}{2\pi f(Range)} \quad ,$$

where f is the measurement frequency and Range is the measurement range selected.

The high frequency AC signal level, Level, is typically kept in the 25 mV range.

When adjusting the step size, Step, it is important to keep the voltage increment within the small signal response of the device. The increment should not be so large that the capacitance reading varies drastically across the voltage increment.

[1] D. K. Schroder, Semiconductor Material And Device Characterization, New York, NY: John Wiley & Sons, Inc., 1990. In particular, pages 50-52 discuss max-min MOS-C capacitance.

[2] E.H. Nicollian and J.R. Brews, MOS Physics and Technology, New York, NY: John Wiley & Sons, 1982. In particular, pages 407-408 discuss the max-min capacitance method and pages 156-172 discuss modeling the high frequency CV curve.

[3] R. Lindner, Bell Sys. Tech. Journal, vol. 41, 1962, p. 803.

[4] Agilent 4284A Precision LCR Meter Operation Manual, Agilent Part Number 04824-9000.

[5] Agilent 4062C/PC/UX Programming Reference, or Agilent 4070 series Programming Reference for BASIC Users.

Data File Format: When using the Plot$ variable to output ASCII data to a file the resulting format will be:

Input parameters	
Output parameters	
Voltage (V)	Capacitance (F)
<data>	<data>

Test Structure

Any MOS capacitor (e.g. dot capacitor from a short loop process or a fully processed capacitor) may be used. If the structures are going to be used for oxide breakdown, see *Introduction to Oxide Integrity* section.

Step-by-Step Action

1) Check for legal input values. If any are out of range, output variables are set to flag values 2.0E+30, parameter error message is displayed, and algorithm exits.
2) Check for legal hardware configuration. If hardware is missing or improperly configured an error message is displayed and the algorithm exits.
3) Connect Agilent 4284A meter as in test configuration diagram.
4) Initialize and set up Agilent 4284A meter (set range, corrections, signal level, frequency, etc.)
6) Lower chuck if prober is in use or prompt operator to remove DUT from test fixture. Then measure the open test fixture offset capacitance.
7) Raise chuck if prober is in use or prompt operator to place DUT in test fixture.
8) Determine polarity required to sweep in the user selected direction based on substrate type.
9) Apply the starting voltage for the sweep.
10) Turn prober light on if prober is in use or prompt operator to apply a light source.

11) Turn prober light off if prober is in use or prompt operator to shield the device from the light source.

12) Measure Cmin and loop until the value settles.

13) Sweep voltage from Start to Stop, performing a voltage and capacitance measurement after waiting Tdelay at each voltage increment Step.

14) If analysis is turned off (Calc$= "N") then set output values to missing, print analysis turned off message, display a plot of the CV Sweep data if plotting is turned on, and exit the algorithm.

15) Correct all measured capacitance values for open test fixture offset value previously measured.

16) If Calc$= "Y", then calculate Nb, Vfb, Vmg, Vth and Tox from CV data.

17) Plot the CV Sweep, marking Vfb, Vmg, and Vth on the graph with the voltage values displayed in the legend. The calculated bulk doping value (Nb) is displayed in the graph title.

18) Return Cox, Cmin, Nb, Vfb, Vmg, and Vth, Tox.

Error Code Summary

This value is assigned to the output variable if an error occurred during algorithm execution:

Value	Meaning
2.0E+30	Input parameter error or meter error
1.5E+30	Analysis not performed
+XE+32	Operating system error "X" occurred during execution of algorithm.

Special Setup Instructions for Agilent 4062UX and Agilent 4070 series

1) Connect the Agilent 4284 Hcur and Hpot connections together through a BNC "T" to the CMH port of the Agilent 4070 series test head or Agilent 4085 switching matrix.

2) Connect the Agilent 4284A Lcur and Lpot connections together through a BNC "T" to the CML port of the Agilent 4070 series test head or Agilent 4085 switching matrix.

Compatibility with Previous Versions

An algorithm, **HFCV_CAP**, is provided to act as a parameter translation shell to maintain compatibility with HP BASIC test programs written for the obsolete **HFCV_CAP**. This algorithm, which calls **HFCV_3_CAP,** is briefly described below. It is highly recommended that test plans be modified to use **HFCV_3_CAP** because of enhanced functionality.

HFCV_CAP

Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Tox	Real	Å	75	10.0~10E3	Design oxide thickness
Dope	Real	cm^{-3}	1E17	1E+11~1E+22	Estimated doping
C_corr	Integer	Meters	0	0~4	Cable length correction
S_cor$	String	——	"N"	"N","Y"	Short correction (Y or N)
O_cor$	String	——	"N"	"N","Y"	Open correction (Y or N)
Start†	Real	Volts	3.3	0~40	Start sweep voltage
Stop†	Real	Volts	3.3	0~40	Stop sweep voltage
Step†	Real	Volts	0.1	0.005~40	Sweep step size
Tstart	Real	Seconds	1	0.01~60	Time to hold start voltage
Tdelay	Real	Seconds	0.05	0.01~60	Delay between each step
Tstop	Real	Seconds	5.0	5~60	Time to hold stop voltage
Freq	Real	Hz	100E+3	20~1E+6	Test frequency
Integ	Integer	——	2	1~3	Meter integration time
Level	Real	Vrms	0.015	0.01~10	Test signal level
Range	Real	Ohms	0	0~1E5	Meter range
Calc$	String	——	"Y"	"N","Y"	Run analysis calculations?
Plot$	String	——	"Y"††	"N","Y"	Plot/Save Data

† Must be specified as a positive number. The algorithm will determine the correct polarities based on the substrate type (N or P).

††A "Y" plots the data on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files*.

.

Device Variables

Var Name	Var Type	Comment/Description
Area	Real	cm^2, Area of active region
Per†	Real	cm, Perimeter of active region
Tty$†	String	(N/P/Al), Top electrode type
Bty$	String	(N/P), Dopant type of bulk region

† Not used by WLR Oxide Algorithms

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Cox	Real	Farads	Maximum capacitance in accumulation
Cmin	Real	Farads	Minimum capacitance in true inversion
Nb	Real	$atoms/cm^3$	Doping concentration
Vfb	Real	Volts	Flatband voltage
Vmg	Real	Volts	Midgap voltage
Vth	Real	Volts	Threshold voltage

Algorithm Syntax

The syntax for calling the algorithm directly is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction to Oxide Integrity* section.

Hfcv_cap (Tox, Dope, C_corr, S_corr$, O_corr$, Start, Stop, Step, Tstart, Tdelay, Tstop,
Freq, Integ, Level, Range, Calc$, Plot$,
Pins(*), Area, Per, Tty$, Bty$,
Cox, Cmin, Nb, Vfb, Vmg, Vth)

Methodology

The algorithm is a translation shell which calls the **HFCV_3_CAP** algorithm to conduct the test. The shell maintains compatibility with existing HP BASIC test programs by removing/adding the following input parameters in the call from **HFCV_CAP** to **HFCV_3_CAP**:

Input parameters:

Tox: Removed, not used in **HFCV_3_CAP.**

Dope: Removed, not used in **HFCV_3_CAP.**

Tstart: Removed, not used in **HFCV_3_CAP**

Tstop: Removed, not used in **HFCV_3_CAP**

Dirswp: Set to 1, not supported in **HFCV_CAP.**

Chktmp: Set to 25, not supported in **HFCV_CAP.**

Output parameters:

No changes.

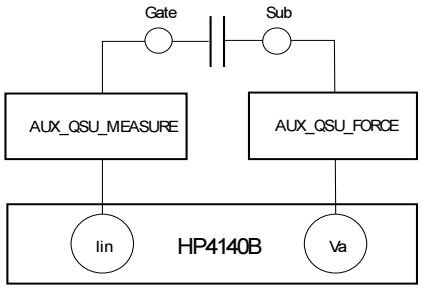
QCV_3_CAP

Measure low frequency of MOS capacitor using quasi-static technique.

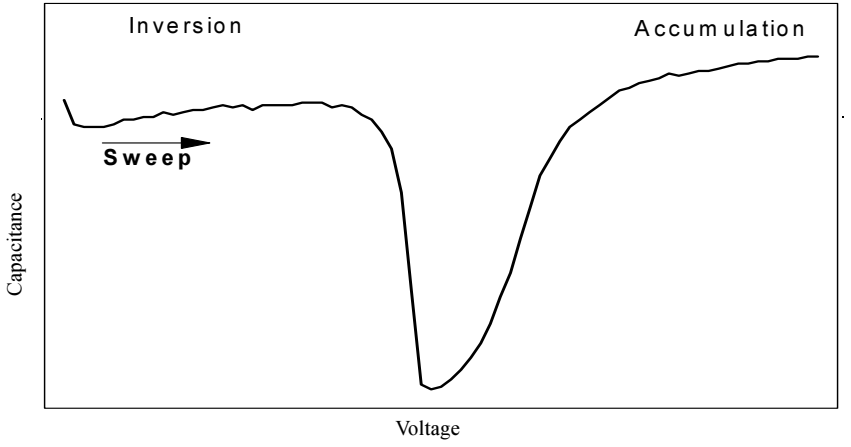
The **QCV_3_CAP** algorithm uses the quasi-static technique to measure the low frequency MOS capacitance. A voltage ramp sweeps the capacitor in a user selectable direction (sweep from inversion is preferred). The current is measured at user defined intervals with the capacitance extracted.

This test configuration requires an Agilent 4140B pA Meter/Voltage Source configured as shown below. See *System Configuration* in the *Installation* chapter.

Test Configuration



Electrical Characteristics



Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Cmax	Real	Farads	1E-9	1E-15~1E-6	Estimate of maximum capacitance†
Start††	Real	Volts	3.3	0~10	Start sweep voltage
Stop††	Real	Volts	3.3	0~10	Stop sweep voltage
Step††	Real	Volts	0.05	0.01~1	Sweep step size
Hold	Real	Seconds	2	0.1~30	Time to hold start voltage
Delay	Real	Seconds	0.5	0.05~30	Delay between each step
Filter	Integer	——	1	0,1	Noise filter 1=ON and 0=OFF
Integ	Integer	——	3	1~3	Sets the integration level 1: Short, 2: Medium, 3: Long
Dirswp	Integer	——	1	1~3	Sweep direction 1: Inversion to accumulation 2: Accumulation to inversion 3: Both directions starting at inversion
Light_on$	String	——	"Y"		Prompt user to cycle light during Hold
Correction$	String	——	"S"	"S","F"	Correction type (full/single)
Plot$	String	——	"Y"†††		Plot/Save Data

† Sets the current measurement range of the Agilent 4140B pA meter. A value too low cause an out of range condition over part of the sweep. A value to high will cause loss of resolution.

†† Must be specified as a positive number. The algorithm will determine the correct polarity based on the bulk type (N or P).

†††A "Y" plots C versus V on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files*.

Device Variables

Var Name	Var Type	Comment/Description
Area	Real	cm^2, Area of active region
Per†	Real	cm, Perimeter of active region
Tty$†	String	(N/P/Al), Top electrode type
Bty$	String	(N/P), Dopant type of bulk region

† Not used by WLR Oxide Algorithms

Algorithm Output Variables:

None

Algorithm Syntax

The syntax for calling the algorithm directly (that is, outside of Agilent SPECS) is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction to Oxide Integrity* section.

Qcv_3_cap(Cmax, Start, Stop, Step, Hold, Delay, Filter, Integ, Dirswp,
Light_on$, Correction$, Plot$,
Pins(*), Area, Per, Tty$, Bty$)

Theory of Operation

If this algorithm is to be used as a part of an oxide stress experiment, then the *Introduction to Oxide Integrity* section provides a general overview.

QCV_3_CAP performs a low frequency capacitance versus voltage measurement on a MOS capacitor using the quasi-static technique. Users unfamiliar with this technique should review references [1,2]. The routine requires the user input value Cmax to set the range of the Agilent 4140B pA meter. To achieve good resolution, Cmax should be measured using the **HFCV_3_CAP** algorithm or calculated from the oxide thickness and capacitor area.

This routine is useful for characterizing process-induced oxide damage as well as performing routine process monitoring. Horizontal shifts in the curve indicate trapped charges, while slope changes result from interface states.

The routine **CV_3_CAP** uses **QCV_3_CAP** and **HFCV_3_CAP** to determine the interface state density (Dit) at the Si-SiO_2 interface.

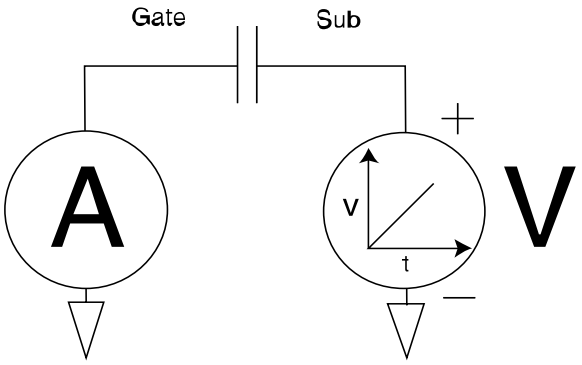
[1] D. K. Schroder, Semiconductor Material And Device Characterization, New York, NY: John Wiley & Sons, Inc., 1990, pages 267-274.

[2] E.H. Nicollian and J.R. Brews, MOS Physics and Technology, New York, NY: John Wiley & Sons, 1982, pages 71-175.

Methodology

The algorithm ramps the voltage on the substrate in order to sweep the device in the user specified direction. The default, sweep from inversion, is recommended and a source of minority carriers (light source) is needed at the start of the sweep. The gate current is measured in user-defined intervals (Delay). A detailed illustration of the voltage ramp applied and the current measured by the Agilent 4140B can be seen on page 3-24, Figure 3-15 in *Agilent 4140B pA Meter/DC Voltage Source Operation and Service Manual*, Agilent Part Number 04140-90021.

A schematic diagram of the quasi-static CV measurement is shown below:



The Agilent 4140A meter output is connected to the substrate rather than the gate to minimize the effects of stray capacitance and noise on the measurement. The current meter is connected to the gate to avoid the stray low impedance paths that are typical at other terminals. Since the voltage source output is applied to the substrate, it causes the gate bias voltage to be the inverse of the source output voltage. The algorithm corrects for this.

The capacitance is calculated from the measured current at each voltage step using the formula:

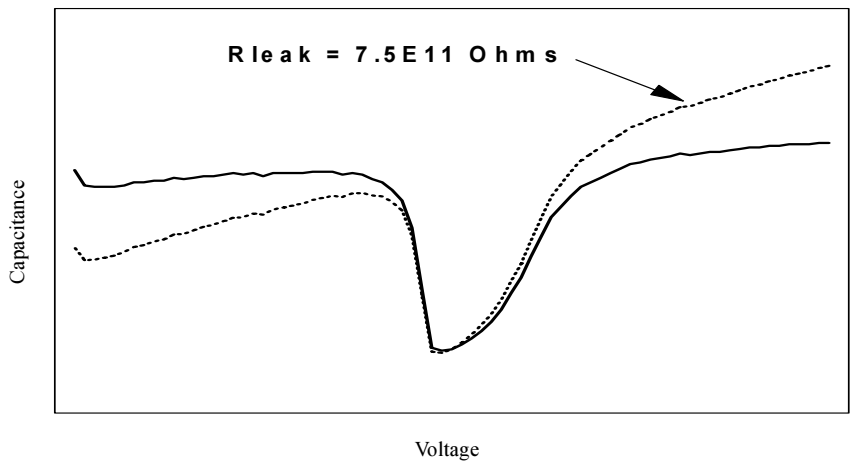
$$C = \frac{I}{dV/dT} \quad ,$$

where I = measured current value, and dV/dT is the linear ramp rate = $\frac{Step}{Delay}$.

Test fixture offset capacitance can be measured and subtracted from the measured capacitance values in one of two ways. If the Correction$ parameter is set to "S", then a single point offset correction is performed. This is the fastest technique. The DUT is disconnected and the open test fixture capacitance is measured at several points about 0. These capacitance values are averaged together as one capacitance measurement and this value becomes the single point capacitance correction which is subtracted from all measured capacitance values in the sweep. Typical fixture offset capacitance values range from 2-6 pF for a switching matrix and probe card.

If Correction$ is set to "F", then a full sweep correction is performed. A complete CV sweep is performed on the test fixture and the measured capacitance values are stored. After the CV is performed, the full sweep offset values are subtracted point-by-point from the measured MOS capacitance values. The full sweep correction is only necessary when an extremely leaky test fixture is being used. A leaky test fixture is evidenced by a large slope in the CV curve, as shown in the following figure:

Quasi-Static CV: Leaky Test Fixture



Cmax is marked on the CV curve to give the user feedback on whether the proper range has been selected.

Caveats and Requirements

When adjusting the step size, Step, it is important to keep the voltage increment within the small signal response of the device. The increment should not be so large that the current reading varies drastically across the voltage increment.

It is also important to use low noise BNC and triax cables and a metal test box to shield the device since low currents may need to be measured.

Data File Format: When using the Plot$ variable to output ASCII data to a file the resulting format will be:

Input parameters	
Output parameters	
Voltage (V)	Capacitance (F)
<data>	<data>

Test Structure

Any MOS capacitor (e.g. dot capacitor from a short loop process or fully processed capacitor) may be used. If the structures are going to be used for oxide breakdown, see "Test Structures" in the *Introduction to Oxide Integrity* section.

Step-by-Step Action

1) Check for legal input values. If any are out of range, output variables are set to flag values 2.0E+30, parameter error message is displayed, and algorithm exits.
2) Check for legal hardware configuration. If hardware is missing or improperly configured an error message is displayed and the algorithm exits.
3) Connect Agilent 4140B meter as in test configuration diagram.
4) Initialize and set up Agilent 4140B meter.
6) Lower chuck if prober is in use or prompt operator to remove DUT from test fixture. Then measure the open test fixture offset capacitance for single point or full sweep correction depending on the value of Correction$ input parameter.
7) Raise chuck if prober is in use or prompt operator to place DUT in test fixture.
8) Determine sweep polarity.
9) Apply the start voltage for Hold seconds. Prompt user to cycle light to supply minority carriers to achieve inversion.
10) Sweep voltage from Start to Stop performing a voltage and current (i.e. capacitance) measurement after waiting Delay at each voltage increment Step.
11) Correct capacitance values according to measured offset and selected correction method.
12) Plot the CV curve, and mark Cmax.
13) Disconnect the meter and reset all test hardware.

Error Code Summary

This value is assigned to the output variable if an error occurred during algorithm execution:

Value	Meaning
2.0E+30	Input parameter error or meter error
+XE+32	Operating system error "X" occurred during execution of algorithm

Special Setup Instructions for Agilent 4070 series

1) This example assumes that AUX_QSU_FORCE is a coax connection to Aux 3, and that AUX_QSU_MEASURE is a triax connection to Aux 1. Refer to the WLR.config file for mapping Aux ports to these names.

2) Connect the Agilent 4140B Va (V OUTPUT) to the AUX3 Force connector of the Agilent 4070 series test head using a standard BNC cable.

3) If the Agilent 16054A option is installed on the Agilent 4140B, set the switch to connect the inner braid (Guard) to ground and connect the triax port to the AUX1 triax port on the Agilent 4070 series test head. If the Agilent 16054A option is not installed, use two triax to coax converters and use coax cable to connect the Agilent 4140B triax port to the triax port on the Agilent 4070 series test head.

Special Setup Instructions for Agilent 4062UX

1) This example assumes that AUX_QSU_FORCE is a coax connection to Aux 2, and that AUX_QSU_MEASURE is a coax connection to Aux 1. Refer to the WLR.config file for mapping Aux ports to these names.

2) Connect the Agilent 4140B Va (V OUTPUT) to the AUX2 connector of the Agilent4085 switch matrix using a standard BNC cable.

3) Connect a standard triax cable to the Agilent 4140B Iin (I INPUT) connection. Connect the other end of this cable to the Agilent 4085 AUX1 connector using a triax to BNC adapter with the two outer conductors (shield and guard) shorted together to ground.

Detailed schematics of these connections can be seen on page 3-45, Figure 3-28 (1-a), and page 3-46, Figure 3-28 (2-a), of the *Agilent 4140B pA Meter/DC Voltage Source Operation and Service Manual.*

Compatibility with Previous Versions

An algorithm, **QCV_CAP**, is provided to act as a parameter translation shell to maintain compatibility with HP BASIC test programs written for the obsolete **QCV_CAP**. This algorithm, which calls **QCV_3_CAP,** is briefly described below. It is highly recommended that test plans be modified to use **QCV_3_CAP** because of enhanced functionality and accuracy.

QCV_CAP

Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Start†	Real	Volts	3.3	0~10	Start sweep voltage
Stop†	Real	Volts	3.3	0~10	Stop sweep voltage
Step†	Real	Volts	0.05	0.01~1	Sweep step size
Hold	Real	Seconds	2	0.1~30	Time to hold start voltage
Delay	Real	Seconds	0.5	0.05~30	Delay between each step
Tox	Real	Å	75	10~1E4	Design oxide thickness
Filter	Integer	——	1	0~1	Noise filter 1=ON and 0=OFF
Correction$	String	——	"S"	"S"~"F"	Correction type (full/single)
Plot$	String	——	"Y"††		Plot/Save Data

† Must be specified as a positive number. The algorithm will determine the correct polarities based on the bulk type (N or P).

††A "Y" plots C versus V on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files*.

Device Variables

Var Name	Var Type	Comment/Description
Area	Real	cm^2, Area of active region
Per†	Real	cm, Perimeter of active region
Tty$†	String	(N/P/Al), Top electrode type
Bty$	String	(N/P), Dopant type of bulk region

† Not used by WLR Oxide Algorithms

Algorithm Output Variables:

None

Algorithm Syntax

The syntax for calling the algorithm directly is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the Introduction section.

Qcv_cap (Start, Stop, Step, Hold, Delay, Tox, Filter, Correction$, Plot$, Pins(*), Area, Per, Tty$, Bty$)

Methodology

The algorithm is a translation shell which calls the **QCV_3_CAP** algorithm to conduct the test. The shell maintains compatibility with existing HP BASIC test programs by removing/adding the following input parameters in the call from **QCV_CAP** to **QCV_3_CAP**:

Input parameters:

Tox: Used to calculate Cmax for input into **QCV_3_CAP.**

Integ: Set to 3, not supported in **QCV_CAP.**

Dirswp: Set to 1, not supported in **QCV_CAP**.

Light_on: Set to N, not supported in **QCV_CAP.**

Output parameters:

No changes.

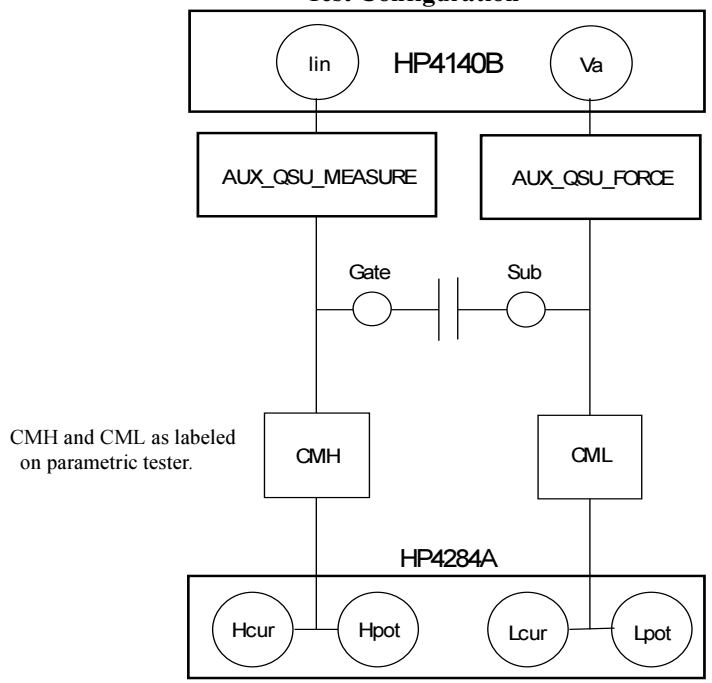
CV_3_CAP

High frequency and quasi-static CV measurements with Dit analysis.

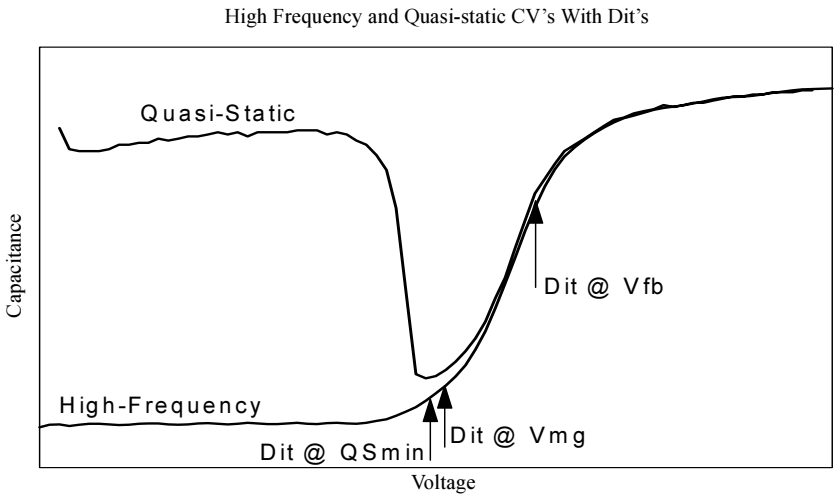
In **CV_3_CAP** a high frequency CV is measured (using **HFCV_3_CAP**), then a quasi-static CV is measured (using **QCV_3_CAP**). The high frequency and quasi-static CV curves are plotted separately, then a plot with both curves overlaid is generated.

The voltages Vfb, Vmg, and Vth are calculated along with the interface state (Dit) value at Vfb, Vmg, and at the minimum on the quasi-static curve (QSmin). These values are plotted on the curve. This test requires an Agilent 4284A Precision LCR Meter with the Power Amp option 001 installed and an Agilent 4140B pA Meter. See *System Configuration* in the *Installation* chapter.

Test Configuration



Electrical Characteristics



Input Variables

Var Name[1]	Var Name 2	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Chktmp	A	Real	°C	27		Chuck temperature
Hc_corr	B	Real	Meters	0	0~4	HF (high frequency) Cable length correction
Hs_corr$	C$	String	——	"N"	"N"~"Y"	HF Short correction (Y or N)
Ho_corr$	D$	String	——	"N"	"N"~"Y"	HF Open correction (Y or N)
V1†	E	Real	Volts	3.3	0~10	HF/QS Start sweep voltage
V2†	F	Real	Volts	3.3	0~10	HF/QS Stop sweep voltage
Vs_h†	G	Real	Volts	0.1	0.005~10	HF Sweep step size
Tdelay	H	Real	Seconds	0.05	0.01~60	HF Step delay time
Freq	I	Real	Hertz	1E6	20~1E6	HF Test frequency
Level	J	Real	Vrms	0.025	0.01~10	HF Test signal level
Range	K	Real	Ohms	0	0~100E3	HF Meter range
Integ	L	Real	——	2	1~3	HF and Qs Meter integration time 1: Short, 2:Medium, 3: Long
Dirswp	M	Real	——	1	1~3	Sweep direction 1: Inversion to accumulation 2: Accumulation to inversion 3: Both directions starting at inversion
Calc$	N$	String	——	"Y"	"N"~"Y"	Run analysis calculations?
Vs_q†	O	Real	Volts	0.05	0.01~1	QS Sweep step size
Hold	P	Real	Seconds	2	0.1~30	QS Time to hold start voltage
Delay	Q	Real	Seconds	0.5	0.05~30	QS Step delay time
Qfilter	R	Real	——	1	0~1	QS Filter
Light_on	S$	String	——	"Y"	"N"~"Y"	Prompt user to cycle light
Q_corr	T$	String	——	"S"	"S","F"	Correction type (full/single)
Plot$	U$	String	——	"Y"††		Plot/Save Data

[1]Variable name under Agilent-ICMS and HP BASIC.

[2]Variable Name under Agilent SPECS.

† Must be specified as a positive number. The algorithm will determine the correct polarities based on the substrate type (N or P).
††A "Y" plots capacitance versus voltage on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files*.

Device Variables

Var Name	Var Type	Comment/Description
Area	Real	cm^2, Area of active region
Per†	Real	cm, Perimeter of active region
Tty$†	String	(N/P/Al), Top electrode type
Bty$	String	(N/P), Dopant type of bulk region

† Not used by WLR Oxide Algorithms

Output Variables

Var Name[1]	Var Name[2]	Var Type	Var Unit	Comment/Description
Cmax	O1	Real	Farads	Maximum capacitance in accumulation
Cmin	O2	Real	Farads	Minimum capacitance in true inversion
Nb	O3	Real	cm^{-3}	Doping concentration
Vfb	O4	Real	Volts	Flatband voltage
Vmg	O5	Real	Volts	Midgap voltage
Vth	O6	Real	Volts	Threshold voltage
Vfb_dit	O7	Real	$cm^{-2}\ eV^{-1}$	D_{it} at flatband voltage
Vmg_dit	O8	Real	$cm^{-2}\ eV^{-1}$	D_{it} at midgap voltage
Cmin_dit	O9	Real	$cm^{-2}\ eV^{-1}$	D_{it} at quasi-static CV minimum capacitance

[1]Variable name under Agilent-ICMS and HP BASIC.

[2]Variable Name under Agilent SPECS.

Algorithm Syntax

The syntax for calling the algorithm directly (that is, outside of Agilent SPECS) is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction* section.

Cv_3_cap(Chktmp, Hc_corr, Hs_corr$, Ho_corr$, V1, V2, Vs_h, Tdelay, Freq,
Level, Range, Integ, Dirswp, Calc$, Vs_q, Hold, Delay, Qfilter,
Light_on$, Q_corr$, Plot$,
Pins(*), Area, Per, Tty$, Bty$,
Cmax, Cmin, Nb, Vfb, Vmg, Vth, Vfb_dit, Vmg_dit, Cmin_dit)

Theory of Operation

If this algorithm is to be used as a part of an oxide stress experiment, then the *Introduction to Oxide Integrity* section provides a general overview.

Users unfamiliar with high frequency and quasi-static capacitance-voltage techniques should review the **HFCV_3_CAP** and **QCV_3_CAP** algorithms. Those unfamiliar with CV analysis techniques, especially extraction of interface trap density Dit, should also review reference [1].

This algorithm performs a high frequency capacitance-voltage (CV) sweep on a MOS capacitor. If the Calc$ parameter is set to "Y" and Dirswp set to 1 or 3 then the routine analyzes the CV data and determines the flatband voltage Vfb, midgap voltage Vmg, threshold voltage Vth, and substrate or bulk doping Nb of the MOS system and plots them on the CV curve. If Calc$ is set to "N" then only the CV sweep curve is plotted.

CV_3_CAP then performs a quasi-static CV sweep on the same MOS capacitor and plots the quasi-static CV curve, followed by a plot of the two curves (HF and QS) overlaid. If Calc$ is set to "Y" then density of interface traps (Dit) are calculated at Vfb, Vmg, and minimum capacitance of the quasi-static CV (QSmin).

This routine is useful for characterizing process-induced oxide damage as well as performing routine process monitoring. Shifts in the Vfb, Vmg or Vth can be used as a measure of charge trapping in the oxide, while changes in Dit quantify the change in interface states.

Methodology

The set-up for the high frequency and quasi-static tests are the same as those described in the test algorithms **HFCV_3_CAP** and **QCV_3_CAP**. The HF curve analysis methodology used to determine Vfb, Vmg, and Nb is also the same as that described in **HFCV_CAP**.

The method of deriving the density of interface states at the minimum capacitance in the quasi-static CV (Dit @ QSmin) is:

1) Determine Cmax (the maximum capacitance in accumulation) from the **HFCV_3_CAP** curve and CLF (the minimum QCV capacitance) and its voltage from the **QCV_3_CAP** curve.

2) Determine CHF (the HFCV capacitance at the voltage corresponding to CLF) by three point Lagrange interpolation.

3) Calculate the capacitor gate area

$$Area = C_{max} \cdot (T_{ox}/\varepsilon_{ox}),$$

where T_{ox} = oxide thickness and ε_{ox} is the permittivity of SiO_2.

4) Calculate Dit at the minimum capacitance, CLF, in the quasi-static curve [1]:

$$D_{it} = \left(\frac{C_{max}}{q \cdot Area}\right)\left[\left(\frac{C_{LF}/C_{max}}{1 - C_{LF}/C_{max}}\right) - \left(\frac{C_{HF}/C_{max}}{1 - C_{HF}/C_{max}}\right)\right] .$$

This technique is also used to calculate Dit @ Vfb and Dit @ Vmg, where CLF is replaced by the capacitance in the quasi-static curve at Vfb or Vmg.

[1] D. K. Schroder, Semiconductor Material And Device Characterization, New York, NY: John Wiley & Sons, Inc., 1990. In particular, pages 267-272 discuss interface trapped charge and the low-frequency (quasi-static) method.

[2] E.H. Nicollian and J.R. Brews, MOS Physics and Technology, New York, NY: John Wiley & Sons, 1982. Chapter 8 discusses extraction of interface trap properties from the capacitance. Pages 331-333 explain the combined high-low frequency method used in this algorithm.

[3] Agilent 4140B pA Meter/DC Voltage Source Operation and Service Manual, Agilent Part Number 04140-90021.

Caveats

The validity of Dit calculated using the above combined high-low frequency method is entirely dependent on both the high frequency and quasi-static curves overlaying one another at Cmax in inversion as modeled on page 251 of [1]. The program adjusts the location of the curves so they overlap in inversion to compensate for offsets due to test instrument effects. The individual curves should be examined for large offsets between the curves which may be due to the presence of mobile ions (which can be characterized using **MBLION_3_CAP or MBLION_SHS**).

Data File Format: When using the Plot$ variable to output ASCII data to a file the resulting format will be:

Input parameters			
Output parameters			
HFCV Voltage(V) Inv - Acc	HFCV Capacitance (F) Inv - Acc	HFCV Voltage(V) Acc - Inv	HFCV Capacitance (F) Acc - Inv
<data>	<data>	<data>	<data>
QCV Voltage(V) Inv - Acc	QCV Capacitance (F) Inv - Acc	__	__
<data>	<data>	__	__
QCV Voltage(V) Acc - Inv	QCV Capacitance (F) Acc - Inv	__	__
<data>	<data>	__	__

Test Structure

Any MOS capacitor (e.g. dot capacitor from a short loop process or fully processed capacitor) may be used. If the structures are going to be used for oxide breakdown, see "Test Structures" in the *Introduction to Oxide Integrity* section.

Step-by-Step Action

1) Check for legal input values. If any are out of range, output variables are set to flag values 2.0E+30, parameter error message is displayed, and algorithm exits.

2) Check for legal hardware configuration. If hardware is missing or improperly configured an error message is displayed and the algorithm exits.

3) Connect Agilent 4284A meter as in Test Configuration diagram.

4) Run high frequency CV sweep as in algorithm flow of HFCV_3_CAP algorithm.

6) Disconnect Agilent 4284 meter and reset all hardware.

7) Connect Agilent 4140B meter as in Test Configuration diagram.

8) Run quasi-static CV sweep as in algorithm flow of QCV_3_CAP.

9) Disconnect Agilent 4140B and reset all hardware.

10) a) If analysis is requested (Calc$="Y"), then plot HFCV and QCV overlaid, calculate all Dit's, plot them on the curve and return all output variables.

b) If analysis is turned off only the HFCV and QCV curves will be plotted overlaid.

Error Code Summary

This value is assigned to the output variable if an error occurred during algorithm execution:

Value	Meaning
2.0E+30	Input parameter error or meter error
1.5E+30	Analysis not performed
9.9E+35	Unable to calculate Dit at this condition
+XE+32	Operating system error "X" occurred during execution of algorithm

Special Setup Instructions for Agilent 4070 series

1) This example assumes that AUX_QSU_FORCE is a coax connection to Aux 3, and that AUX_QSU_MEASURE is a triax connection to Aux 1. Refer to the WLR.config file for mapping Aux ports to these names.

2) Connect the Agilent 4140B Va (V OUTPUT) to the AUX3 Force connector of the Agilent 4070 series test head using a standard BNC cable.

3) If the Agilent 16054A option is installed on the Agilent 4140B, set the switch to connect the inner braid (Guard) to ground and connect the triax port to the AUX1 triax port on the Agilent 4070 series test head. If the Agilent 16054A option is not installed, use two triax to coax converters and use coax cable to connect the Agilent 4140B triax port to the triax port on the Agilent 4070 series test head.

Special Setup Instructions for Agilent 4062UX

1) This example assumes that AUX_QSU_FORCE is a coax connection to Aux 2, and that AUX_QSU_MEASURE is a coax connection to Aux 1. Refer to the WLR.config file for mapping Aux ports to these names.

2) Connect the Agilent 4140B Va (V OUTPUT) to the AUX2 connector of the Agilent 4085 switch matrix using a standard BNC cable.

3) Connect a standard triax cable to the Agilent 4140B Iin (I INPUT) connection. Connect the other end of this cable to the Agilent 4085 AUX1 connector using a triax to BNC adapter with the two outer conductors (shield and guard) shorted together to ground.

Detailed schematics of these connections can be seen on page 3-45, Figure 3-28 (1-a), and page 3-46, Figure 3-28 (2-a), of [3].

Compatibility with Previous Versions

An algorithm, **CV_CAP**, is provided to act as a parameter translation shell to maintain compatibility with HP BASIC test programs written for the obsolete **CV_CAP**. This algorithm, which calls **CV_3_CAP,** is briefly described below. It is highly recommended that test plans be modified to use **CV_3_CAP** because of enhanced functionality.

CV_CAP

Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Tox	Real	Å	75	10~10E3	Design oxide thickness
Dope	Real	cm^{-3}	1E17	1E11~1E22	Estimated doping
Hc_corr$	Integer	Meters	0	0~4	HF (high frequency) Cable length correction
Hs_corr$	String	——	"N"	"N"~"Y"	HF Short correction (Y or N)
Ho_corr$	String	——	"N"	"N"~"Y"	HF Open correction (Y or N)
Q_corr$	String	——	"S"	"S"~"F"	QS (quasi-static) Single/Full correction
V1†	Real	Volts	3.3	0~10	HF/QS Start sweep voltage
V2†	Real	Volts	3.3	0~10	HF/QS Stop sweep voltage
Hv3†	Real	Volts	0.1	0.005~10	HF Sweep step size
Qv3†	Real	Volts	0.05	0.01~1	QS Sweep step size
Ht1	Real	Seconds	1	0.01~60	HF Time to hold start voltage
Qt1	Real	Seconds	2	0.1~30	QS Time to hold start voltage
Ht2	Real	Seconds	0.05	0.01~60	HF Step delay time
Qt2	Real	Seconds	0.5	0.05~30	QS Step delay time
Ht3	Real	Seconds	5	5~60	HF Time to hold stop voltage
Freq	Real	Hertz	100E3	20~1E6	HF Test frequency
Integ	Integer	——	2	1~3	HF Meter integration time
Level	Real	Vrms	0.015	0.01~10	HF Test signal level
Range	Real	Ohms	0	0~100E3	HF Meter range
Qfilter	Integer	——	1	0~1	QS Filter
Calc$	String	——	"Y"	"N"~"Y"	Run analysis calculations?
Plot$	String	——	"Y"††		Plot/Save Data

† Must be specified as a positive number. The algorithm will determine the correct polarities based on the substrate type (N or P).
††A "Y" plots C versus V on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files*.

Device Variables

Var Name	Var Type	Comment/Description
Area	Real	cm^2, Area of active region
Per†	Real	cm, Perimeter of active region
Tty$†	String	(N/P/Al), Top electrode type
Bty$	String	(N/P), Dopant type of bulk region

† Not used by WLR Oxide Algorithms

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Cmax	Real	Farads	Maximum capacitance in accumulation
Cmin	Real	Farads	Minimum capacitance in true inversion
Nb	Real	$atoms/cm^{-3}$	Doping concentration
Vfb	Real	Volts	Flatband voltage
Vmg	Real	Volts	Midgap voltage
Vth	Real	Volts	Threshold voltage
Vfb_dit	Real	$cm^{-2}\ eV^{-1}$	D_{it} at flatband voltage
Vmg_dit	Real	$cm^{-2}\ eV^{-1}$	D_{it} at midgap voltage
Cmin_dit	Real	$cm^{-2}\ eV^{-1}$	D_{it} at quasi-static CV minimum capacitance

Algorithm Syntax

The syntax for calling the algorithm directly is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the Introduction section.

Cv_cap (Tox, Dope, Hc_corr, Hs_corr$, Ho_corr$, Q_corr$, V1, V2, Hv3, Qv3, Ht1, Qt1, Ht2, Qt2, Ht3, Freq, Integ, Level, Range, Qfilter, Calc$, Plot$, Pins(*), Area, Per, Tty$, Bty$,
Cmax, Cmin, Nb, Vfb, Vmg, Vth, Vfb_dit, Vmg_dit, Cmin_dit)

Methodology

The algorithm is a translation shell which calls the **CV_3_CAP** algorithm to conduct the test. The shell maintains compatibility with existing HP BASIC test programs by removing/adding the following input parameters in the call from **CV_CAP** to **CV_3_CAP**:

Input parameters:

Tox: Removed, not used in **CV_3_CAP**

Dope: Removed, not used in **CV_3_CAP**

Ht1: Removed, not used in **CV_3_CAP**

Ht3: Removed, not used in **CV_3_CAP**

Dirswp: Set to 1, not supported in **CV_CAP**

Light_on: Set to N, not supported in **CV_CAP**

Chktmp: Set to 27, not supported in **CV_CAP**

Output parameters:

No changes.

MBLION_3_CAP

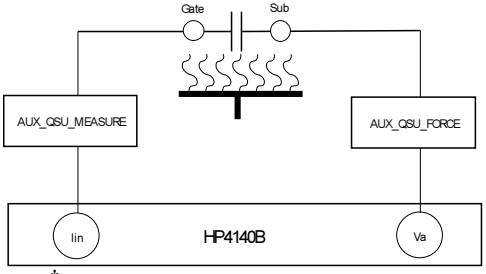
Uses a triangular voltage sweep technique at elevated temperature to measure mobile ions.

The **MBLION_3_CAP** algorithm uses the triangular current-voltage technique to measure the mobile ion density of a MOS capacitor. Since all measurements are made at an elevated temperature without repeatedly heating or cooling the chuck, test time is significantly reduced.

This test configuration requires an Agilent 4140B pA Meter/Voltage Source.

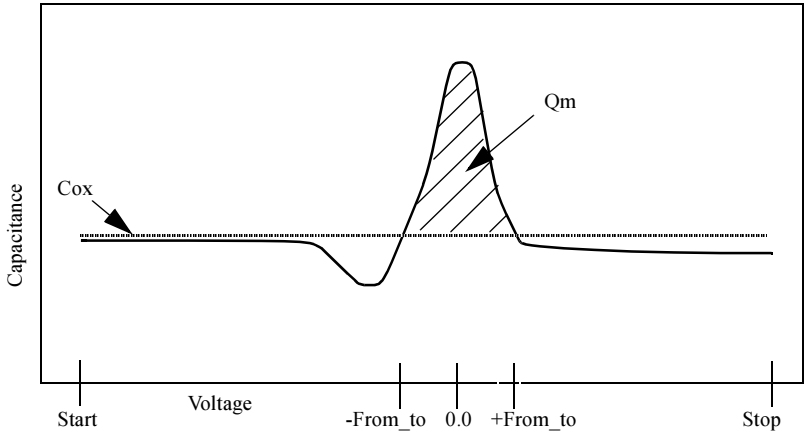
The test configuration requires a thermochuck. See *System Configuration* in the *Installation* chapter.

Test Configuration



†Temperature must be set before beginning test of single wafer or lot.

Electrical Characteristics



Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Cmax	Real	Farads	1E-9	1E-15~1E-6	Maximum capacitance from HFCV_3_CAP (Cox)
Start†	Real	Volts	5	0~10	Start sweep voltage
Stop†	Real	Volts	5	0~10	Stop sweep voltage
Step	Real	Volts	0.05	0.01~1	Sweep step size
From_to	Real	Volts	2	0.01~10	Integrate data from/to this voltage
Hold	Real	Seconds	1	0.1~30	Time to hold start voltage
Delay	Real	Seconds	0.5	0.05~30	Delay between each step
Filter	Integer	——	1	0~1	Noise filter (On or Off)
Integ	Integer	——	3	1~3	Integration 1:Short, 2:Medium,3:Long
Dirswp	Integer	——	3	1~3	Sweep direction 1: Inversion to accumulation 2: Accumulation to inversion 3: Both directions starting at inversion
Light_on$	String	——	"Y"		Prompt user to cycle light during Hold
Correction$	String	——	"S"	"S"~"F"	Correction type (full/single)
Plot$	String	——	"Y"††		Plot/Save Data

† Must be specified as a positive number. The algorithm will determine the correct polarities based on the bulk type (N or P).

††A "Y" plots the capacitance versus voltage data on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files*.

Device Variables

Var Name	Var Type	Comment/Description
Area	Real	cm^2, Area of active region
Per†	Real	cm, Perimeter of active region
Tty$†	String	(N/P/Al), Top electrode type
Bty$	String	(N/P), Dopant type of bulk region

† Not used by WLR Oxide Algorithms

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Nm	Real	Ions/cm^2	Number of mobile ions

Algorithm Syntax

The syntax for calling the algorithm directly (that is, outside of Agilent SPECS) is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction to Oxide Integrity* section.

Mblion_3_cap(Cmax, Start, Stop, Step, From_to, Hold, Delay, Filter, Integ,
Dirswp, Light_on, Correction$, Plot$,
Pins(*), Area, Per, Tty$, Bty$,
Nm)

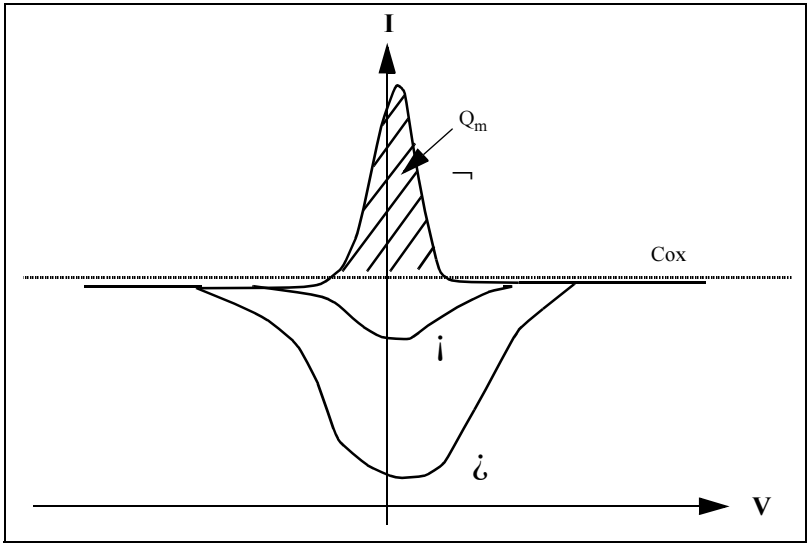
Theory of Operation

The *Introduction to Oxide Integrity* section provides a general overview.

This routine is useful for characterizing and identifying mobile ions introduced into the integrated circuit manufacturing process. This mobile charge in the silicon dioxide is primarily due to the ionic impurities sodium (Na+), potassium (K+) or lithium (Li+). **MBLION_3_CAP** can be used to perform quick qualitative tests at relatively low temperatures as well as quantitative tests at higher temperatures. It can be used also to differentiate between mobile ions such as Na+ and K+. The elevated temperatures (150 ^{o}C

to 300 ^{o}C) used to detect the presence of mobile ions are applied using a hot chuck. However, it is only necessary to heat up the chuck once before beginning a test. An entire wafer or lot of wafers may be measured without repeatedly heating or cooling the chuck at each test site. This saves a significant amount of test time and is the key advantage of **MBLION_3_CAP**.

This algorithm uses a triangular voltage sweep technique to measure the density of mobile ions in a MOS device [1]. A linear voltage sweep is applied to a capacitor and the resulting current is measured. At room temperature, the "mobile" charge is immobile and the measured current is directly related to the capacitance of the MOS device. The result is a quasi-static CV curve (see also **QCV_3_CAP**) shown in the figure on the next page. At an elevated temperature (150 ^{o}C to 300 ^{o}C) and without mobile ions, a similar curve is seen. However, the dip in the C-V curve is reduced due to temperature dependence of the intrinsic carrier concentration and other parameters. In the presence of mobile ions, an obvious increase in the maximum measured current (capacitance) is noted. In particular, a "peaked" curve is observed. The area under the peaked portion of the curve above the line (Cox) is proportional to the mobile ion charge.



Capacitance-voltage response to linear voltage sweep of MOS capacitor at room temperature ¿, at high temperature with no mobile ions ¡, and high temperature with mobile ions ¬.

In order to obtain quantitative results, several assumptions are made. It is assumed that all the mobile charges are initially located at the polysilicon (or metal)-oxide interface. This can be achieved using a sufficiently long Hold time at the Start voltage and performing the test at a high enough temperature. Furthermore, it is assumed that all the mobile charge are located at the oxide-silicon interface at the Stop voltage. This is achieved if a slow enough ramp rate and high enough temperature are used. If a qualitative indication of the presence of mobile ions is needed, then a lower temperature and faster ramp rate (faster test time) can be used.

Methodology

MBLION_3_CAP ramps the voltage on the substrate to sweep the device in a user selectable direction and measures the capacitance of the device at the with a user defined ramp rate (Step/Delay). A detailed illustration of the voltage ramp applied and current measured by the Agilent 4140B can be seen on page 3-24, Figure 3-15 in [2].

The test time required by this technique can be reduced by increasing test temperature and ramp rate and reducing voltage sweep range. If Na+ (or Li+) is the major concern, then the sweep range can usually be reduced to -1 V to +1 V.

The **QCV_3_CAP** documentation also provides additional comments that are also applicable for this routine.

[1] D.K. Schroder, Semiconductor Material and Device Characterization, New York, NY: John Wiley & Sons, Inc., 1990, pages 264-267.

[2] Agilent 4140B pA Meter/DC Voltage Source Operation and Service Manual, Agilent Part Number 04140-90021.

Caveats and Requirements

Since this mobile ion technique is similar to quasi-static CV techniques, the caveats listed in **QCV_3_CAP** are also appropriate here. In addition, quantitative require a ramp rate low enough and temperature high enough such that all mobile charge can drift through the oxide.

The range of the Agilent 4140B is set based on 2*Cmax. For very high mobile ion concentrations in thick dielectrics the measured current may be outside this range. If this occurs, the range can be extended by increasing Cmax and correcting the Nm calculation for the fixed offset Cmaxnew-Cmax.

Data File Format: When using the Plot$ variable to output ASCII data to a file the resulting format will be:

Input parameters	
Output parameters	
Voltage (V)	Current (A)
<data>	<data>

Test Structure

Any MOS capacitor (e.g. dot capacitor from a short loop process or fully processed capacitor) may be used. If the structures are going to be used for oxide breakdown, see "Test Structures" in the *Introduction to Oxide Integrity* section.

Step-by-Step Action

1) Check for legal input values. If any are out of range, output variables are set to flag values 2.0E+30 and algorithm exits.
2) Connect gate and substrate to Agilent 4140B as illustrated.
3) Determine polarity based on substrate type.
4) Initialize Agilent 4140B meter.
5) Lower chuck if prober is in use or prompt operator to remove DUT from test fixture. Then measure the open test fixture offset capacitance for single point or full sweep correction depending on the value of Correction$ input parameter.
6) Raise chuck if prober is in use or prompt operator to place DUT in test fixture.
9) Apply the start voltage for Hold seconds. Prompt user to cycle light to supply minority carriers to achieve inversion.
10) Sweep voltage from Start to Stop performing a voltage and current (i.e. capacitance) measurement after waiting Delay at each voltage increment Step.

11) Sweep voltage from Start to Stop after holding start voltage Hold seconds, performing a voltage and current measurement after waiting Delay at each voltage increment Step.

12) Correct current (or capacitance) values according to measured offset and selected correction method.

13) Plot the CV Sweep, calculating and marking Cox design on the plot.

10) Calculate and return Nm.

11) Disconnect the meter and reset all test hardware.

Error Code Summary

This value is assigned to the output variable if an error occurred during algorithm execution:

Value	Meaning
2.0E+30	Input parameter error
1.5E+30	Analysis not performed
+XE+32	Operating system error "X" occurred during execution of algorithm

Special Setup Instructions for Agilent 4070 series

1) This example assumes that AUX_QSU_FORCE is a coax connection to Aux 3, and that AUX_QSU_MEASURE is a triax connection to Aux 1. Refer to the WLR.config file for mapping Aux ports to these names.

2) Connect the Agilent 4140B Va (V OUTPUT) to the AUX3 Force connector of the Agilent 4070 series test head using a standard BNC cable.

3) If the Agilent 16054A option is installed on the Agilent 4140B, set the switch to connect the inner braid (Guard) to ground and connect the triax port to the AUX1 triax port on the Agilent 4070 series test head. If the Agilent 16054A option is not installed, use two triax to coax converters and use coax cable to connect the Agilent 4140B triax port to the triax port on the Agilent 4070 series test head.

Special Setup Instructions for Agilent 4062UX

1) This example assumes that AUX_QSU_FORCE is a coax connection to Aux 2, and that AUX_QSU_MEASURE is a coax connection to Aux 1. Refer to the WLR.config file for mapping Aux ports to these names.

2) Connect the Agilent 4140B Va (V OUTPUT) to the AUX2 connector of the Agilent 4085 switch matrix using a standard BNC cable.

3) Connect a standard triax cable to the Agilent 4140B Iin (I INPUT) connection. Connect the other end of this cable to the Agilent 4085 AUX1 connector using a triax to BNC adapter with the two outer conductors (shield and guard) shorted together to ground.

Detailed schematics of these connections can be seen on page 3-45, Figure 3-28 (1-a), and page 3-46, Figure 3-28 (2-a), of [2].

Compatibility with Previous Versions

An algorithm, **MBLION_CAP**, is provided to act as a parameter translation shell to maintain compatibility with HP BASIC test programs written for the obsolete **QCV_CAP**. This algorithm, which calls **MBLION_3_CAP,** is briefly described below. It is highly recommended that test plans be modified to use **MBLION_3_CAP** because of enhanced functionality and accuracy.

MBLION_CAP

Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Start†	Real	Volts	5	0~10	Start sweep voltage
Stop†	Real	Volts	5	0~10	Stop sweep voltage
Step	Real	Volts	0.05	0.01~1	Sweep step size
From_to	Real	Volts	2	0.01~10	Integrate data from/to this voltage
Hold	Real	Seconds	1	0.1~30	Time to hold start voltage
Delay	Real	Seconds	0.5	0.05~30	Delay between each step
Tox	Real	Å	110	10.0~10E3	Design oxide thickness
Filter	Integer	——	1	0~1	Noise filter (On or Off)
Correction$	String	——	"S"	"S"~"F"	Correction type (full/single)
Plot$	String	——	"Y"††		Plot/Save Data

† Must be specified as a positive number. The algorithm will determine the correct polarities based on the bulk type (N or P).

††A "Y" plots the data on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files*.

Device Variables

Var Name	Var Type	Comment/Description
Area	Real	cm^2, Area of active region
Per†	Real	cm, Perimeter of active region
Tty$†	String	(N/P/Al), Top electrode type
Bty$	String	(N/P), Dopant type of bulk region

† Not used by WLR Oxide Algorithms

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Nm	Real	Ions/cm^2	Number of mobile ions

Algorithm Syntax

The syntax for calling the algorithm directly is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction to Oxide Integrity* section.

Mblion_cap(Start, Stop, Step, From_to, Hold, Delay, Tox, Filter,
Correction$, Plot$,
Pins(*), Area, Per, Tty$, Bty$,
Nm)

Methodology

The algorithm is a translation shell which calls the **MBLION_3_CAP** algorithm to conduct the test. The shell maintains compatibility with existing HP BASIC test programs by removing/adding the following input parameters in the call from **MBLION_CAP** to **MBLION_3_CAP**:

Input parameters:

Tox: Used to calculate Cmax for input into **MBLION_3_CAP.**

Integ: Set to 3, not supported in **MBLION_CAP.**

Dirswp: Set to 1, not supported in **MBLION_CAP**.

Light_on: Set to N, not supported in **MBLION_CAP.**

Output parameters:

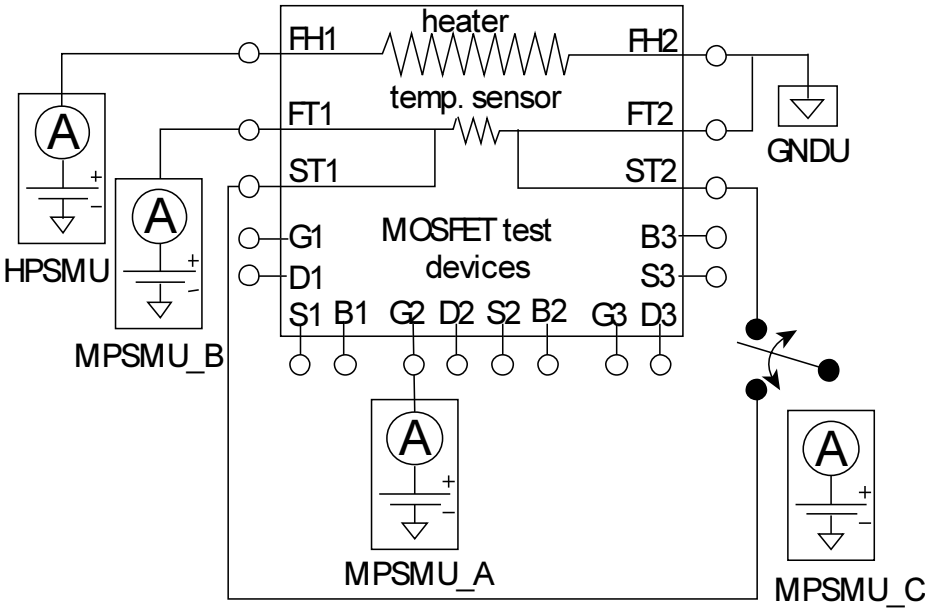
No changes.

MBLION_SHS

Performs a biased-temperature or temperature stress on a Self-Heated Structure.

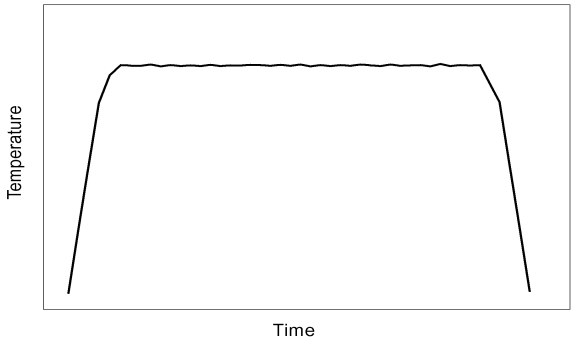
The algorithm maintains a constant heater temperature for a user-specified time. Up to three MOSFET thresholds can be measured before and after the stress. When used with the MBL test structure, this algorithm performs a biased-temperature stress to detect mobile ions. The stress-induced shift in parasitic MOSFET threshold voltages is used to detect of mobile ions.

Test Configuration — Stress

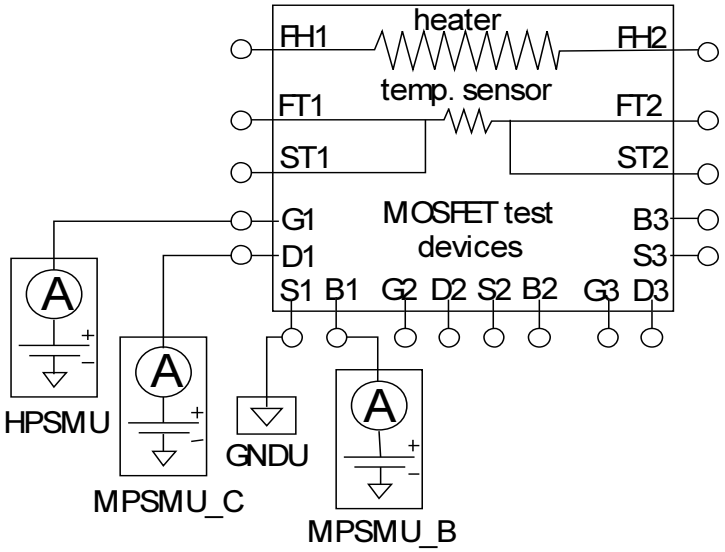


Note: MPSMU_C is shared during the stress as shown.

Electrical Characteristic



Test Configuration — Measurement



Note: This test configuration is used if MOSFET device pins are nonzero. Measurement configuration is repeated for each device specified (e.g. G2, D2, S2, B2 for device 2).

Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min ~ max)	Comment/Description
Temp	Real	^{o}C	400	0 ~ 550	Heater test temperature
Time	Real	Sec	10	1 ~ 1E6	Stress time at heater temperature
Dirs	Integer	--	1	1,2,3,4	Stress type 1: I, V negative w.r.t. ground followed by positive I,V stress 2: I, V positive stress only 3: I, V negative stress only 4: I, V positive followed by negative I, V stress
Beta	Real	1/^{o}C	3.55E-3	1E-3 ~ 1E-1	Temperature Coefficient of Resistance of metal thermometer
Tb	Real	^{o}C	27		Temperature at which Beta is defined
Idth	Real	Amps	1E-7	1E-12 ~ 100E-6	Current at which threshold voltage is defined.
Vds	Real	Volts	0.1	-5 ~ 5	Drain-to-source voltage during threshold measurement(s)
Vg1	Real	Volts	0	-200 ~ 200	Minimum Vgs for threshold search+
Vg2	Real	Volts	100	-200 ~ 200	Maximum Vgs for threshold search+
Vg3	Real	Volts	30	0 ~ 100	Absolute value Gate to substrate voltage during stress *
Chuck_tmp	Real	^{o}C	27		Ambient temperature or chuck temperature
Plot$	String	--	"Y"		Plot/Save Data

* Sign is based on Dirs variable

+ Vg1 must not equal Vg2

Device Variables

There are no device variables.

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Vt1	Real	Volts	Initial threshold of device 1
Vt2	Real	Volts	Initial threshold of device 2
Vt3	Real	Volts	Initial threshold of device 3
D1r	Real	Volts	Difference initial and post stress reverse stress device 1
D2r	Real	Volts	Difference initial and post stress reverse stress device 2
D3r	Real	Volts	Difference initial and post stress reverse stress device 3
D1f	Real	Volts	Difference initial and post stress forward stress device 1
D2f	Real	Volts	Difference initial and post stress forward stress device 2
D3f	Real	Volts	Difference initial and post stress forward stress device 3
R_o	Real	Ohms	Initial thermometer resistance at Chuck_tmp
Rh_o	Real	Ohms	Initial heater resistance at Chuck_tmp
Beta_h	Real	1/^{o}C	Approximate heater Beta at Chuck_tmp
Q_h	Real	^{o}C/Watt	Heater thermal resistance – slope of temperature vs. power curve

Mode +	Real	--	Heater stress failure code 00000 - Normal measurement -1 - No temperature stress 00007 - Failure before expected set time 00020 - Failure during ramp down to Chuck_tmp 00500 - Heater open 05000 - Heater shorted 00100 - Thermometer open 00900 - could not measure Kelpvin resistance 01000 - Thermometer shorted 20000 - Exceeded current range of SMU 10000 - Exceeded voltage range of SMU

+ Multiple failure modes may occur. In this case the modes are added together (e.g., mode=600 is from 00100 - Thermometer open and 00500 - Heater open).

Algorithm Syntax

The syntax for calling the algorithm directly (this is, outside of Agilent SPECS using HP BASIC) is Inputs, Pins Array, Device Parameters and Outputs.

MBLION_SHS (Temp, Time, Dirs, Beta, Tb, Idth, Vds, Vg1, Vg2, Vg3,
Chuck_tmp,
Plot$,
Pins(*),
Vt1, Vt2, Vt3, D1r, D2r, D3r, D1f, D2f, D3f, R_o, Rh_o, Beta_h, Q_h,
Mode)

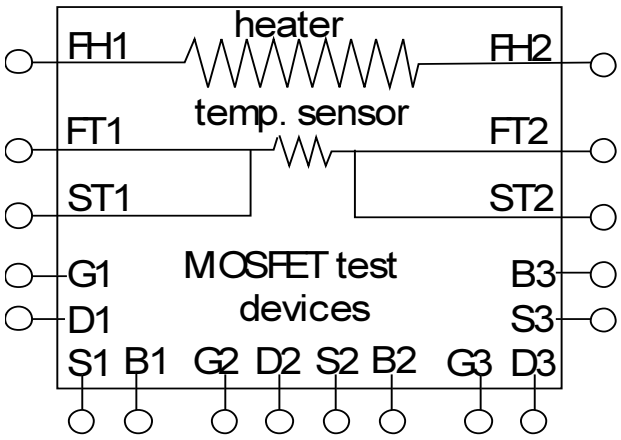
Test Structure Definition

MBLION_SHS is meant to be used with the MBL self-heated mobile-ion test structure available as a separate product from Agilent. This algorithm requires a conducting heater element (e.g. polysilicon or polycide) and a Kelvin measurement temperature sensor. A conducting material with a linear temperature coefficient of resistance is needed for the temperature sensor. Threshold measurement of up to 3 MOSFET devices is also supported. The heater must be sized accordingly to the proper thermal test structure design and to match the capability of the instrument.

This algorithm can be used with other properly designed self-heated structures (e.g., non-volatile memory retention structures). The temperature stress provided in this algorithm could be combined with a user written mea-

surement algorithm in an Agilent SPECS testplan or separate HP BASIC algorithm.

Self-Heated Device Schematic



Self-Heated Device Specifications

Terminal Connections	Position in Pins Array	Comment/Description	
Fh1	1	High side of Heater	Heater
Fh2	2	Low side of Heater	
St1	3	High Sense of Temperature Sensor	Thermo-meter
St2	4	Low Sense of Temperature Sensor	
Ft1	5	High Force of Temperature Sensor	
Ft2	6	Low Force of Temperature Sensor	
Mg2	7	Optional Gate Bias during Stress*	
Mg3	8	Optional Gate Bias during Stress*	

G1	9	Gate MOSFET Device 1 (optional)**	Device 1
D1	10	Drain Device 1	
S1	11	Source Device 1	
B1	12	Bulk Device 1	
Sub1	13	Substrate Contact Device 1***	
Chk1	14	Wafer Chuck Device 1***	
G2	15	Gate MOSFET Device 2 (optional)**	Device 2
D2	16	Drain Device 2	
S2	17	Source Device 2	
B2	18	Bulk Device 2	
Sub2	19	Substrate Contact Device 2***	
Chk2	20	Wafer Chuck Device 2***	
G3	21	Gate MOSFET Device 3 (optional)**	Device 3
D3	22	Drain Device 3	
S3	23	Source Device 3	
B3	24	Bulk Device 3	
Sub3	25	Substrate Contact Device 3***	
Chk3	26	Wafer Chuck Device 3***	

*Mg2 and Mg3 provide pins for the voltage Vg3 to be applied during the stress.
**If not defined (i.e. set equal to 0), then algorithm will not make threshold voltage measurements.
***Not necessary to define.

Theory of Operation

The Introduction to OX section provides a general overview of oxide integrity. Since self-heating is used, the user may also wish to consult Introduction to EM section as well as **ISO_4_EMR** algorithm for further details on self-heating, temperature measurement and control. For mobile ions, the user can consult the algorithm **MBLION_3_CAP**.

Mobile ions are ionic charges (e.g. Na+ or K+) which are introduced unintentionally during IC processing. This may be due to human contact, impure chemicals, residue from photoresist, contaminated metal targets, etc. As the name implies, the ionic species are mobile enough at higher operating temperatures that they can cause IC reliability problems. For example, mobile

ions can cause reduction in device isolation, formation of parasitic transistors or increased diode leakage.

The ionic current in oxides is proportional to

$$J_{ion} \propto \frac{E}{T} e^{(-E_a)/(kT)} \qquad (1)$$

where E = electric field across the oxide; T = oxide temperature; E_a = activation energy for ion diffusion; k = Boltzmann's constant. Equation 1 shows that there is a large temperature dependence and a small electric field dependence. Nevertheless, it requires both temperature and electric field to move mobile ions through an oxide.

A classical mobile ion test is the Bias-Temperature Stress (BTS). The traditional BTS measures a large area capacitor on a hot chuck. The flatband voltage is measured before stress. A negative (or positive) voltage is applied to the gate and then the chuck temperature is increased to a high temperature. The temperature is maintained for at least several minutes while the ionic charge moves to the gate (or SiO_2-Si interface). The temperature is cooled down to room temperature to freeze the mobile ions in place. The flatband voltage is then measured. The test is repeated for a positive (or negative) voltage and the flatband voltage is again measured. The change in flatband voltage is assumed due to the presence of mobile ions. If the temperature is high enough and the stress time is long enough, then the flatband voltage shift can be used to calculate the mobile ion density.

The traditional BTS test is not suitable for production monitoring because of the slow hot chuck temperature ramp, high temperature hot chuck and large area capacitor. In addition, probe tips will slip during the temperature ramp which makes the test more complicated.

The Triangular Voltage Sweep (TVS) technique was developed to overcome the problems associated with the changing chuck temperature. TVS requires a hot chuck to heat the oxide. However, the chuck temperature remains constant during the stress and the voltage is changed. This techniques is implemented in the **MBLION_3_CAP** algorithm. While the problems associated with the changing chuck temperatures are eliminated, the resolution of the technique is limited by size of the capacitor structure.

The **MBLION_SHS** algorithm performs a biased-temperature stress using a self-heated test structure. The temperature is provided by an on-chip heating

element (typically polysilicon or silicide) and is measured accurately using a resistance thermometer. There are advantages to using a self-heating structure to perform a BTS: no hot chuck, no slow hot chuck temperature ramp, no probe tip slippage, higher temperature stress (i.e. faster test) and smaller test structure. When using with the MBL test structure (See Test Structure Definition), tests faster than 30 seconds and at greater than 400 ^{o}C may be obtained.

Methodology

MBLION_SHS performs a constant temperature and voltage stress using an on-chip heater structure. The heater temperature is controlled using a resistance thermometer (usually metal). The temperature coefficient of resistance of the thermometer is used to determine the heater temperature. The algorithm performs a controlled ramp up to Temp, remains at this temperature for the user specified Time and then performs a controlled ramp down to Chuck_tmp. Active real-time instrument feedback is maintained between the temperature sensor and heater during the ramp and constant temperature stress.

The **MBLION_SHS** algorithm provides a separate voltage for biased-temperature stress. The voltage Vg3 is applied before the temperature ramp and is removed after the ramp down to Chuck_tmp. The polarity of voltage stress is set by the user input Dirs. This variable also controls the applied current direction and voltage polarity applied to the heater. For example, Dirs=2 provides a positive voltage stress with respect to ground. This positive stress will force positively charged ions to the SiO_2-Si interface.

Support is also provided for a bi-directional BTS. If Dirs=1 (or 4) a negative (positive) polarity temperature stress is followed by a positive (negative) temperature stress. For each direction of stress, Temp is maintained for the user-input Time. This bi-directional stress is particularly useful for detecting and characterizing mobile ions.

The algorithm supports threshold measurement of up to three MOSFETs. The change in device threshold before and after the BTS is used to detect mobile ions. The threshold measurement utilizes a binary search to find constant current threshold voltage at Idth with Vds applied. If a device(s) is not specified (through the Pins(*) variable), then the threshold(s) are not measured. If Mg2 or Mg3 is not specified, then Vg3 is not applied during the temperature stress. So, it is possible to measure the threshold of a device but not apply a voltage bias during the temperature structure. Similarly, a voltage bias could be applied during the temperature stress without measuring a threshold voltage.

The initial threshold voltages listed in the output variables list is the threshold before a particular temperature stress. With Dirs=1, D1f is the difference between the post reverse stress value (NOT the initial Vt1 value) and the post-forward-stress threshold voltage. A positive voltage temperature stress is called a forward stress (hence the f notation) and a negative voltage stress is called a reverse stress (r notation). From this, the effective threshold voltage shift due to mobile ions is D1f provided several caveats are met. The temperature and stress time must be high enough to move all the mobile ions to Poly-SiO_2 interface during the negative biased-temperature stress and to reverse the process during the positive biased-temperature stress.

Testing Notes: To accurately measure the temperature, the temperature coefficient of resistance (Beta) of the conducting thermometer must be measured. Beta can easily be measured using the algorithm **TCR_3_EMR**. The same Beta used in EM testing can be used for **MBLION_SHS**.

While an active hot chuck is desirable for **MBLION_SHS**, it is not absolutely necessary. While variation in room temperature will change the heater temperature (see **ISO_4_EMR**), it is of second order importance provided that the stress temperature is high enough to move the charge from one interface to the next in the user specified time. This can be verified by performing a characterization of the test structure of interest and looking for the temperature which causes a change in the voltage. Ideally, this would be performed with a contaminated sample, but this is not necessary. For the MBL test structure, temperatures in the range on 400 ^{o}C to 500 ^{o}C are sufficient.

The user should maintain temperature high enough to perform a rapid test. However, if too high a temperature is used the metal temperature sensor may not accurately reflect the temperature and hence the software may not be able to control the heater temperature. This is not a problem with the software algorithm, but may be the result of an assumed (not measured) temperature coefficient of resistance or that the metal thermometer is no longer working. Recall that the melting point of aluminum is around 660 ^{o}C (over 800 ^{o}C for Copper), while the melting point for polysilicon is over 1000 ^{o}C. Test temperature should be kept well below the melting point of the thermometer (e.g., 100 ^{o}C).

This algorithm, along with the MBL structure, can detect low levels of mobile ions. The resolution is determined by the oxide thickness and resolved threshold voltage. For typical dielectrics, mobile ion densities on the order of 0.5E9 cm^{-2} can be detected.

MPSMU_C is shared between senses of the metal thermometer. This reduces the number of stress SMU's needed to four.

Data File Format: When using the Plot$ variable to output ASCII data to the a file the resulting format will be:

Input parameters			
Output parameters			
Time (sec)	Current (mA)	Resistance (Ohm)	Temperature (^{o}C)
<data>	<data>	<data>	<data>

Step-by-Step Action

1. Check for legal input values. If any are out of range, output variables are set to flag values 2.0E30 and algorithm exits
2. Connect SMUs as in measurement Test Configuration diagram
3. If MOSFET device 1 is present, check for leaky device by applying Vds and measuring Id current. If exceeds Idth, then set outputs 1E-30 and algorithm exits.
4. If device 1 through device 3 are present, measure threshold(s) at Idth using binary search If at least one device is good (|Vt| < 200V) then continue.
5. Connect SMUs as in stress Test Configuration diagram.
6. Check initial resistances of thermometer and heater.
7. Determine current and stress type (bias polarity) for heater ramp to *Temp*.
8. Apply Vg3 with proper polarity to Device 2 and 3 (if Mg2 or Mg3 defined).
9. Ramp heater to target temperature in approximately equal temperature steps.
10. Maintain constant temperature for user specified time (*Time*).
11. Ramp down temperature to *Chuck_tmp* in approximately equal temperature steps.

12. Disconnect Vg3 from Device 2 and 3.
13. If *Dirs* = 1 or 4, repeat steps 2 through 12 with opposite polarity stress.
14. Return Vt1, Vt2, Vt3, D1r, D2r, D3r, D1f, D2f, D3f, R_o, Rh_0, Beta_h, Q_h and Mode.

Error Code Summary

This value is assigned to the output variable if an error occurred during algorithm execution:

Value	Meaning
1E-99	Device consider shorted during measurement
1E-30	Device leaky (>Idth) from drain to source (or substrate) in device 1
2.0E+30	One or more input parameters are out of range
2.1E+30	Unable to determine threshold, increase Vg2
2.2E+30	Unable to determine threshold, lower Vg1 or increase difference between Vg1 and Vg2
2.3E+30	Unable to determine threshold, abnormal entry from threshold search, consider changing search limit Vg1 and/or Vg2
1E+31	Heater or thermometer failure
1E+36	Device open or Vt above present Vg2 (increase Vg2)
1.5E+30	Analysis not performed
+XE+32	Operating system error "X" occurred during execution of algorithm

5 Metal and Interconnect Reliability

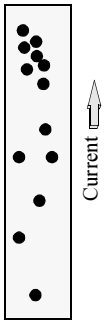
Introduction to Electromigration (EM)

This section describes a general overview of the WLR Electromigration algorithms.

High current densities induce migration of metallic atoms

Voids or hillocks form in the metal structure

Results in increased resistance or short to adjacent metal structure

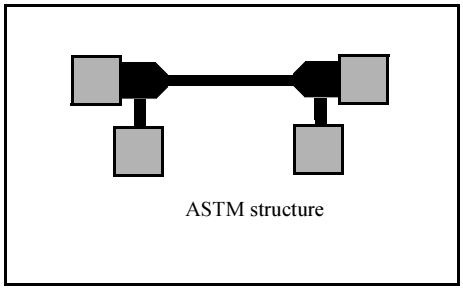


Electromigration Algorithms

Name	Device Type	Comment/Description
KELRES_3	EMRES	Kelvin resistance measurement
TCR_3	EMRES	JEDEC JESD33 compliant temperature coefficient of resistance measurement
TVP_3	EMRES	Thermal resistance measurement
ISO_4	EMRES	JEDEC JESD61 compliant constant temperature electromigration test
BEM_3	EMRES	Breakdown-energy-of-metals test
SWEAT_3	EMRES	JEDEC JEP119 compliant constant time-to failure electromigration test
CONST_I_3	EMRES	Constant current electromigration test
ISO	EMRES	Obsolete constant temperature EM test

Test Structure Definition

The Electromigration algorithms are defined for device type EMRES (ElectroMigration RESistor). The device type definition supports both the linear ASTM-like structure and the more geometrically complex structures, such as via, contact and SWEAT structures. Because a high degree of accuracy is critical, four point measurements (Kelvin) are used.



EMRES Device Specifications

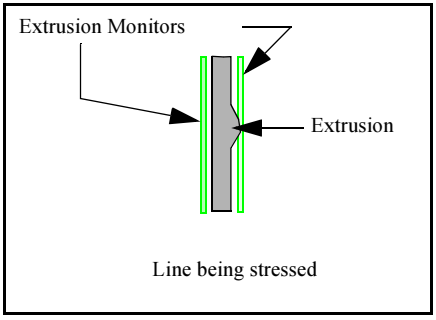
Terminal Connections	Position in Pins Array	Comment/Description
F1	1	High Side of Structure
F2	2	Low Side of Structure
S1	3	High Sense
S2	4	Low Sense
E1	5	Optional Extrusion Monitor
E2	6	Optional Extrusion Monitor

Var Name	Var Type	Comment/Description
Sweat[1]	Real	1: SWEAT structure 0: non-SWEAT (ASTM-like linear strip)

1. The Sweat parameter is only used in obsolete routines supported for compatibility to existing HP BASIC test programs.

Extrusion Monitor

The EMRES device type defines an optional extrusion monitor which is typically a metal line parallel to or above/below the metal line under test. The monitor can be used to check for shorts caused by the line under test extruding through the dielectric as in the figure below.



Theory of Operation

The wafer level EM algorithms measure the time-to-failure (TTF) of an electromigration (EM) test structure when Joule heating is used to attain the target temperature. The algorithms are based on Black's Equation:

$$TTF = \frac{A}{j^{n}} e^{E_a/kT}$$

where

A = Constant that depends on the material and stripe geometry ($s - \left(\frac{A}{cm^2}\right)^n$)

n = Dimensionless constant (typically 2)

E_a =Activation energy (eV)

k =Boltzmann's constant (8.617×10^{-5} eV/K)

T =Temperature in Kelvin.

j =Current density (A/cm^2)

Characterizing Beta and Q

Beta, the temperature coefficient of resistance, and Q, the thermal resistance, provide the foundation for EM testing. Beta and Q can be process dependent so it is recommended Beta and Q are measured for each lot and in some instances each wafer. The **ISO_3_EMR** and **BEM_3_EMR** algorithms measure Q in-situ for every structure tested and thus eliminates the need for a separate measurement of Q. However, the routines **SWEAT_3_EMR** and **CONST_I_3_EMR** require Q which can be obtained from **executing TVP_3_EMR**. A sample size of at least 5 is recommended for a good estimate of the mean and standard deviation for Q. Beta is needed for all EM routines, and because experience shows that Beta is very uniform over a wafer, only a few measurements are necessary.

Monitoring the Temperature

The temperature is monitored via the structure resistance through the relation:

$$Temp\_now = \frac{R\_\mathrm{stress} - R\_\mathrm{o}}{Beta \times R\_\mathrm{o}} + T\_\mathrm{amb}$$

where

Beta = Temperature Coefficient of Resistance at the chuck temperature (obtained from algorithm **TCR_3_EMR**)

R_stress = Resistance of the structure while under current stress

R_o = Initial resistance at the chuck temperature

T_amb = Chuck temperature.

As electromigration-induced damage causes changes in the initial resistance at the chuck temperature (R_o) of the structure, the structure temperature is in doubt once EM induced damage starts to occur. This adds an element of

uncertainty in the temperature which is common to all EM measurement approaches where Joule heating is used to raise the structure temperature, no matter how slight the temperaure increase might be.

Determining the Cross-sectional Area

The SWEAT test requires measurement of the current density in order to operate. Also, the constant current test (CONST_I_EMR) may use current density. The current density, J, is derived from the applied current, I, and the cross-sectional area of the narrowest portion of the structure by the relation:

$$J = \frac{I}{Area}$$

Two approaches are provided for calculating the cross-sectional area. The first assumes that the experimenter knows the film thickness and the narrowest width of the structure: the cross-sectional area is simply the product of the two. The second assumes that the structure is long with uniform cross-section (ASTM-like), and the cross-sectional area is calculated by the following:

$$Area = \rho\frac{L}{R\_o}$$

where ρ is the resistivity of the metal at the chuck temperature

(2.82×10^{-6} Ω-cm for aluminum at 25°C)

L is the length of the structure (cm)

R_o is the initial resistance of the structure (Ω)

This approach has the advantage that process induced variations in cross-sectional area from structure to structure can be compensated by measuring R_o. This approach can alsobe used for more complicated structures with commensurately more complicated analysis.

Adjusting for SWEAT Structure

When the SWEAT structure is used, the algorithms must adjust the target temperature to compensate for the spatial variation in temperature within the SWEAT structure. That is done through use of the parameter Alpha (used in the **SWEAT_EMR, XISO_EMR, BEM_EMR**, and **CONST_I_EMR** algorithms) which relates the peak temperature (T_peak) to the temperature obtained through resistance thermometry (T_avg) in a SWEAT structure.

A simple calculation for the ratio of the change in temperature of the narrow region of the SWEAT structure to the change in mean temperature of the total structure due to Joule heating is [1]:

$$Alpha = \frac{N_{sn}}{N_{sn}^2 + N_{sw}^2}(N_{sn} + N_{sw})$$

where N_{sn} is the number of squares in the narrow region
N_{sw} is the number of squares in the wide region.

This calculation is quick and simple but suffers from **very large errors** due to the simplifying assumptions used. These errors grossly underestimate the peak temperature. The rationale for the calculation is given in [1]. A much more accurate, and complicated, approach is described in [2].

[1] B. J. Root and T. Turner,"Wafer Level Electromigration Tests for Production Monitoring," Proceedings of the 1985 International Reliability Physics Symposium, pp.100-107, 1985.

[2] C. R. Crowell, C. C. Shih, and V. Tyree, "Simulation and Testing of Temperature Distribution and Resistance Versus Power for Sweat and Related Joule-Heated Metal-On-Insulator Structures," Proceedings of the 1990 International Reliability Physics Symposium, pp.37-43, 1990.

Initial Trial Voltage

All of the EM algorithms start with a measurement of the ambient resistance of the structure. The voltage applied to measure resistance must be small enough such that the applied current induces insignificant Joule heating. In order to efficiently find as large a forcing voltage value as possible that does not heat up the structure, a good initial voltage is necessary. For a straight line structure, the following relation can be used:

The Resistance is $R = \frac{\rho L}{A}$

The Current is $I = JA$

The Voltage is $V = IR$

$$V = \frac{\rho L}{A} JA = \rho LJ \qquad \textbf{(Equation 1)}$$

where ρ =Resistivity (2.8×10^{-6} Ω-cm for aluminum)

L =Length (cm)

J =Peak Current Density (A/cm^2).

Notice that the voltage does not depend on the cross sectional area. Simple thermal calculations show that a current density on the order of 4×10^5 A/cm^2 does not cause significant heating (only an increase of about 0.2 °C). For a standard straight line ASTM electromigration structure, where L = 0.08 cm, the trial voltage is calculated as the following:

$$V = 2.82 \times 10^{-6} \text{ohm-cm} \times 0.08 \text{cm} \times 4 \times 10^5 \text{A/cm}^2 = 0.07 \text{volts}$$

This value is rounded up to 0.1 V as the initial trial voltage (V_cool).

Note: For structures other than aluminum, or with a length different than 0.08 cm, Equation 1 should be used to recalculate the voltage. If the structure is not a uniform cross-section line, the thermal calculations are much more difficult, and trial and error may be the best approach.

Note: All of the electromigration algorithms automatically search for the maximum force voltage that does not heat the structure. However, a good initial voltage guess results in a faster test and less stress on the structure.

Test Structure Design Rules

All EM tests require a Kelvin (four terminal) measurement of resistance. A properly designed EM test structure is necessary for accurate measurement. The structure should have voltage taps at the ends of the structure and the

force lines should be at least twice the minimum width of the metal structure under test. Measurements can be made on any structure geometry from simple uniform lines over flat oxide (ASTM structure) to complicated structures over underlying severe topology or lines containing vias and contacts.

The structure may have extrusion monitors, which are typically minimum pitch metal lines running parallel to the structure. The extrusion monitor is used to check for shorts due to electromigration induced hillock growth. Sometimes an Mi-1/Mi+1 plate under/over Mi is used as a vertical extrusion monitor.

The parameter Alpha is needed to translate resistance measurements to peak temperature measurements for SWEAT structures. It is important to note that the more complicated the structure (such as SWEAT), the more uncertain the peak temperature, regardless of the sophistication of the thermal analysis. Since EM time-to-failure depends exponentially on temperature, small uncertainties in temperature lead to large uncertainties in the time-to-failure. This should be carefully weighed when making the decision on which structure to build and test.

Contact Structures

The electromigration algorithms allow the user to select the polarity of the forced voltages and currents with respect to ground through the parameter Xtype. This feature is needed to test active contact structures where a positive stress is needed for n-type contacts and a negative stress is required for p-type contacts. The default is positive stress.

Test Approach

Our experience shows that the best wafer-level electromigration results are obtained using structures based upon ASTM Standard F1259 along with the JEDEC compliant **ISO_3_EMR** algorithm. This recommendation pertains to structures fabricated with no underlying topography and more complicated structures that contain steps, vias or contacts.

Applicable Test Structures

For users of PDQ-FAB WLR test structures, a separate product available from Agilent, the following table lists PDQ-WLR software algorithm recommended for use with each test structure. Other WLR algorithms, though not recommended, could be used such as SWEAT, constant current and BEM.

PDQ-FAB WLR Metallization Test Structures

Structure	Structure Name	Primary Applicable WLR Algorithms
Metal Line Minimum Width Electromigration Structure. Also includes CD line width and sheet resistance structure	EM_A_M1..N EM_CBR_M1..N	ISO_4 TCR_3 KEL_RES_3
Grain Boundary Electromigration	EM_B_M1..N	ISO_4 TCR_3 KELRES_3
Via Voiding and Electromigration	EM_VIA_A EM_VIA_B EM_VIA_C	ISO_4 TCR_3 KELRES_3
Contact Integrity (Spiking, coverage, etc) & Electromigration	EM_CONT_P EM_CONT_N	ISO_4 TCR_3 KELRES_3
Metal Step Coverage	EM_STEP_A_M1..MN EM_STEP_B_M1..MN	ISO_4 TCR_3 KELRES_3
Electromigration over thin oxide to reduce test time	EM_T_A_M1 EM_T_B_M1 EM_CBR_T_M1	ISO_4 TCR_3 KELRES_3

The next table lists some of the possible failure mechanisms detected using the PDQ-FAB WLR test structures and software.

Electromigration Test Structures and Failure Mechanisms

Test Structure Description	Failure Mechanisms	Tests
Electromigration (EM)	Grain Structure; Etch Uniformity; Stress Voids; Corrosion Voids; Metal Deposition; Dielectric Deposition; Target Doping; Deposition Temperature; Sputter Contamination; Dielectric Strength; Dielectric Thickness; Dielectric Interface; Thermal Process; EM Parameters (Ea, n, A); EM Lifetime (Quantitative)	Isothermal, BEM, SWEAT
EM Step, Contact & Via	Step Coverage; Contact Coverage; Contact Uniformity; Barrier Coverage; Via Coverage; Via Uniformity; Line Width; Line Thickness; Grain Structure; Metal Composition	Isothermal, TCR

Inputting Maxro into Agilent SPECS

A real array is required to input Maxro into ISO_4_EMR and BEM_3_EMR. Agilent SPECS test plans will accommodate this, provided the Agilent SPECS framework has been prepared as detailed in Chapter 1, or if you are using the PDQ-WLR plug framework under Agilent SPECS 2.4 or later. The example that follows assumes that you have chosen to use a Tag Variable array called WLR_ISO_Maxro, as was provided by the sample framework with your distribution.

There are two test plan steps required to satisfy the ISO_4_EMR and BEM_3_EMR inputs:

- Specify the three target values for Maxro.
- Input the array into the target PDQ-WLR algorithm.

The Agilent SPECS Test Plan Navigator presents a window with two middle columns called "Algorithm" and "Input". The following example shows the lines required to set Maxro to its default values and use it for ISO_4_EMR.

Algorithm	Input
LET	WLR_ISO_Maxro[1]=1
LET	WLR_ISO_Maxro[2]=5
LET	WLR_ISO_Maxro[3]=10
ISO_4_EMR_XL	Maxro=WLR_ISO_Maxro

Note that the "Maxro=WLR_ISO_Maxro" statement may come as a result of using the Agilent SPECS Assist graphical user interface for algorithm inputs. Assist will expect the name of an array defined in the current framework to satisfy the Maxro input. ***Remember to select a framework that defines a Maxro array when using test plans that implement ISO_4_EMR and BEM_3_EMR tests. Note that this also applies to ISO_3_EMR.***

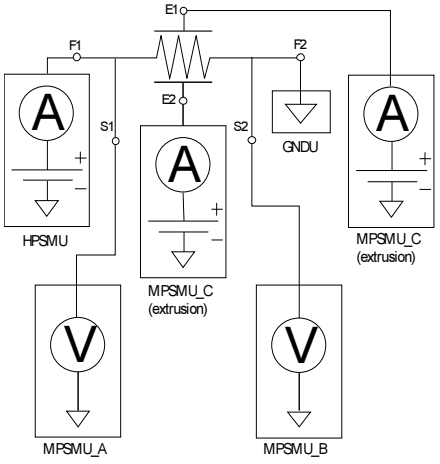
KELRES_3_EMR

Measures the resistance of Kelvin-type electromigration test structures.

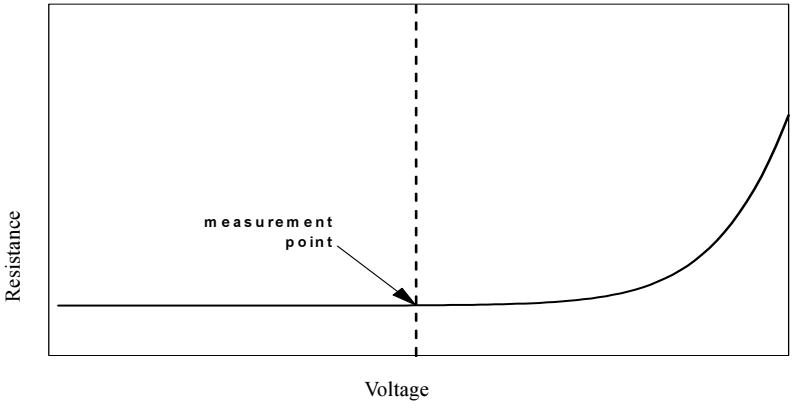
This **KELRES_3_EMR** algorithm forces an initial test voltage and subsequently reduces voltage until a level is found which will cause insignificant Joule heating of structure.

Optional extrusion monitors are supported.

Test Configuration



Electrical Characteristics



Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
V_cool	Real	Volts	100E-3	1E-3~2	Initial trial voltage to measure resistance without heating

Device Variables

Var Name	Var Type	Comment/Description
None		

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Resist	Real	Ω	Calculated resistance
Ires	Real	mA	Average measured current
Vres	Real	Volts	Average forced voltage
Mode	Real	——	Failure code 00001 - short to extrusion monitor 1 00010 - short to extrusion monitor 2 00100 - structure open 00900 - could not measure Kelvin resistance 01000 - structure shorted

Algorithm Syntax

The syntax for calling the algorithm directly (that is, outside of Agilent SPECS) is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction* section.

Kelres_3_emr(V_cool, Plot$,
Pins(*),
Resist, Ires, Vres, Mode)

Theory of Operation

The *Introduction to Electromigration* section provides a general overview.

KELRES_3_EMR is a routine which performs 4-terminal resistance measurements on electromigration and other conductor test structures.

If current density is too great during the resistance measurement a misleading result will occur due to Joule heating. In order to avoid this problem **KELRES_3_EMR** automatically adjusts the forced voltage such that a resistance measurement is made without significantly heating the EM structure.

Test Structure

See "Test Structures" in the *Introduction to Electromigration* section.

Step-by-Step Action

1) Check for legal input values. If any are out of range, output variables are set to flag values 2.0E+30 and algorithm exits.
2) Connect SMUs as in Test Configuration diagram (Extrusion monitors are used if non-zero in device pin-out).
4) Determine forcing voltage, using V_cool as an initial guess, which will not cause heating in the device under test.
5) Measure resistance.
6) Return Resist, Ires, Vres, and Mode.

Error Code Summary

This value is assigned to the output variable if an error occurred during algorithm execution:

Value	Meaning
2.0E+30	One or more input parameters are out of range
1.5E+30	Analysis not performed
1.0E+31	DUT is bad
+XE+32	Operating system error "X" occurred during execution of algorithm

Compatibility with Previous Versions

An algorithm, **KELRES_EMR**, is provided that act as a parameter translation shell to maintain compatibility with HP BASIC test programs written for the obsolete **KELRES_EMR**. This algorithm, which calls **KELRES_3_EMR**, is briefly described below.

KELRES_EMR

Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
V_cool	Real	Volts	100E-3	1E-3~2	Initial trial voltage to measure resistance without heating

Device Variables

Var Name	Var Type	Comment/Description
Sweat	Real	1: SWEAT structure 0: non-SWEAT (ASTM-like linear strip)

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Resist	Real	Ω	Calculated resistance
Ires	Real	mA	Average measured current
Vres	Real	Volts	Average forced voltage
Mode	Real	——	Failure code 00001 - short to extrusion monitor 1 00010 - short to extrusion monitor 2 00100 - structure open 00900 - could not measure Kelvin resistance 01000 - structure shorted

Algorithm Syntax

The syntax for calling the algorithm directly is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the Introduction section.

Kelres_emr (V_cool, Plot$,
Pins(*), Sweat,
Resist, Ires, Vres, Mode)

Methodology

The algorithm is a translation shell which calls the **KELRES_3_EMR** algorithm to conduct the test. The shell maintains compatibility with existing HP BASIC test programs by removing/adding the following input parameters in the call from **KELRES_EMR** to **KELRES_3_EMR**:

Input parameters:

Sweat: Removed, not used in **KELRES_3_EMR.**

Output parameters:

No changes.

TCR_3_EMR

Measure the Temperature Coefficient of Resistance.

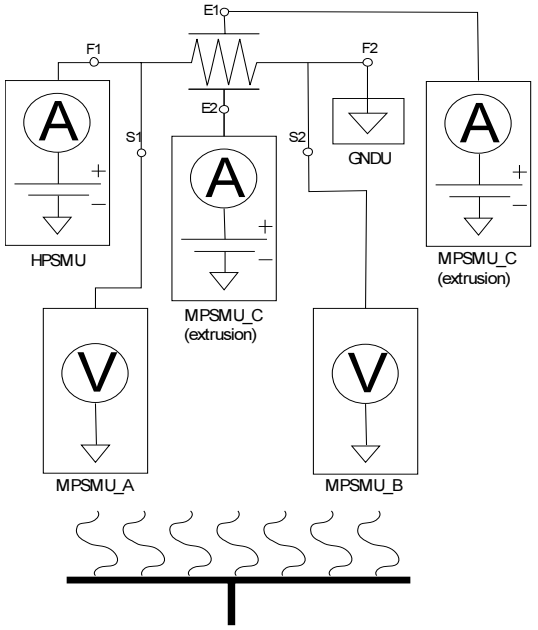
The resistance of an electromigration test structure is measured as a function of temperature in compliance with JEDEC Standard JESD33.

Optional extrusion monitors are supported.

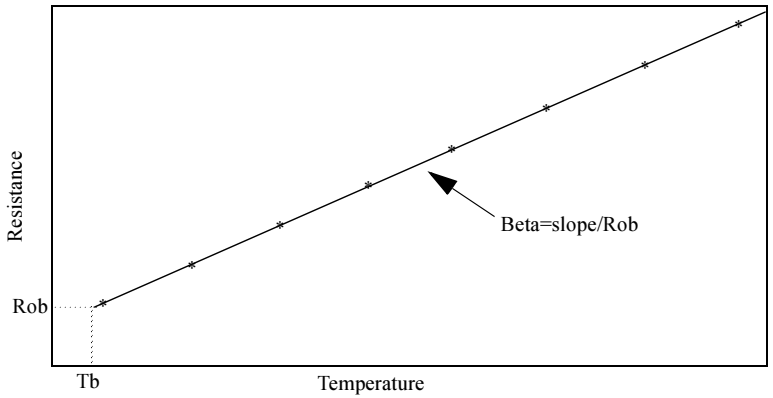
The test configuration requires a thermochuck. See *System Configuration* in the *Installation* chapter.

Test Configuration

Note: MPSMU_C is shared during stress.



Electrical Characteristics



Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
V_cool	Real	Volts	100E-3	1E-3~2	Initial trial voltage to measure resistance without heating
Soak	Real	Sec	300	1~1000	Soak time at temperature
Lowrtmp	Integer	^oC	29	0~100	Lower limit for temperature range
Upprtmp	Integer	^oC	100	60~300	Upper limit for temperature range
Ntemps	Integer	——	4	4~60	Total number of at-temperatures measurements
Plot$	String	——	"Y"††		Plot/Save Data

††A "Y" plots resistance versus temperature on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files*.

Device Variables

Var Name	Var Type	Comment/Description
None		

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Beta	Real	1/°C	Linear temperature coefficient of resistance
Tb	Real	°C	Temperature at which Beta is defined. (same as Lowrtmp)
Rob	Real	Ω	Resistance measured at Lowrtmp
Rfinal	Real	Ω	Post-test resistance measurement at Tb
Corcoef	Real	——	Correlation Coefficient
Intcpt	Real	Ω	Y intercept of Resistance vs. Temperature curve
Mode	Real	——	Failure code 00001 - short to extrusion monitor 1 00010 - short to extrusion monitor 2 00100 - structure open 00900 - could not measure Kelvin resistance 01000 - structure shorted

Algorithm Syntax

The syntax for calling the algorithm directly (that is, outside of Agilent SPECS) is Inputs, Pins Array, Device Parameters, and Outputs. Device information can be found in the *Introduction to Electromigration* section.

Tcr_3_emr(V_cool, Soak, Lowrtmp, Upprtmp, Ntemps, Plot$,
Pins(*),
Beta, Tb, Rob, Rfinal, Corcoef, Intcpt, Mode)

Theory of Operation

The *Introduction to Electromigration* section provides a general overview.

Beta, the temperature coefficient of resistance, is a key parameter in wafer-level electromigration (EM) testing and is used to determine stripe

temperature from stripe resistance measurements during a test. Since EM lifetimes depend exponentially on temperature, accurate knowledge of the temperature through Beta is critical for accurate measurements.

Beta relates the initial resistance measurement, Rob, at the initial chuck temperature, Tb, to the resistance R_t at some temperature (T) by the relation:

$$R(T) = Rob \times (1 + Beta(T - Tb))$$

Beta can be process dependent so it is important to measure Beta for each lot, and in some instances, each wafer. Beta is generally very well behaved and a single sample per wafer may be all that is required.

Ideally, Beta should be measured over a temperature range that includes the temperature to be used for EM testing. Since most hot chucks and probe cards are limited in maximum temperature, Beta should be measured over the range from ambient to the maximum chuck or probe card temperature.

The program defaults assume a chuck capable of reaching 100 ^{o}C. At least 4 discrete temperatures spread uniformly over the range should be used in measuring Beta.

Beta is referenced to some temperature, usually ambient, denoted as Tb in the program. Throughout the EM routines, ambient denotes a chuck temperature set slightly above room temperature in order to remove room temperature variations from the measurements. It is generally good practice to use the same "ambient" temperature for all measurements. Tb is set to the first (lowest) of the user entered temperatures in the program.

The parameter Soak sets the dwell time at each temperature before the resistance measurement is made. Soak is an important parameter since, because of the chuck's thermal mass, it can take several minutes for the wafer to reach the temperature indicated by the chuck thermocouple. The best way to characterize the thermal response of the chuck, and hence determine the soak time needed, is to perform a characterization using a surface contact thermocouple on a scrap wafer.

Testing notes: Instability of metal resistance (due to improperly annealed metal for example) can result in large errors in Beta. The ambient resistance is measured at the end of the Beta test to assure that no resistance shifts have occurred. Rob and Rfinal should be the same, within the repeatability of the

instruments.

Data File Format: When using the Plot$ variable to output ASCII data to a file the resulting format will be:

Input parameters	
Output parameters	
Temperature (C)	Resistance (Ω)
<data>	<data>

Test Structure

See "Test Structures" in the *Introduction to Electromigration* section.

Step-by-Step Action

1) Connect SMUs as in Test Configuration diagram (Extrusion monitors are used if non-zero in device pin-out).
2) Set thermochuck to temperature Tb.
3) Determine forcing voltage, using V_cool as an initial guess, which will not cause heating in DUT.
4) Loop for Ntemps number of temperatures:
   a) Set thermochuck to temperature and wait for Soak time.
   b) Measure resistance 3 times (in both forward and reverse polarity).
   c) Calculate average resistance.
   d) Store resistance at temperature.
5) Rob = resistance at initial temperature.
6) Set thermochuck to temperature Tb; measure Rfinal.
7) Perform a least squares fit of resistance vs. temperature curve (if the correlation coefficient is less than the JEDEC standard of 0.999 then set output variable Corcoef to error flag and continue with normal termination).
8) Calculate Beta.
9) Return chuck to room temperature.

10) Return Beta, Tb, Rob, Rfinal, Corcoef, and Intcpt.

Error Code Summary

This value is assigned to the output variable if an error occurred during algorithm execution:

Value	Meaning
1.0E+30	Thermochuck has failed to reach temperature
2.0E+30	One or more input parameters are out of range
1.0E+31	DUT is bad
2.0E+31	Corcoef variable set to this value if correlation coefficient of fit is less than 0.999
1.5E+30	Analysis not performed
+XE+32	Operating system error "X" occurred during execution of algorithm.

Compatibility with Previous Versions

Two algorithms, **TCR_EMR** and **XTCR_EMR** are provided that act as parameter translation shells to maintain compatibility with HP BASIC test programs written for the obsolete **TCR_EMR** and **XTCR_EMR**. These algorithms, which call **TCR_3_EMR**, are briefly described below.

TCR_EMR

Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
V_cool	Real	Volts	100E-3	1E-3~2	Initial trial voltage to measure resistance without heating
Lowrtmp	Integer	°C	29	0~100	Lower limit for temperature range
Upprtmp	Integer	°C	100	60~300	Upper limit for temperature range
Ntemps	Integer	——	4	4~60	Total number of at-temperatures measurements
Plot$	String	——	"Y"††		Plot/Save Data

††A "Y" plots the resistance versus temperature on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files*.

Device Variables

Var Name	Var Type	Comment/Description
Sweat	Real	1: SWEAT structure 0: non-SWEAT (ASTM-like linear strip)

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Beta	Real	1/K	Linear coefficient of resistance
Tb	Real	°C	Temperature at which Beta is defined. (same as Lowrtmp)
Rob	Real	Ω	Resistance measured at Lowrtmp
Rfinal	Real	Ω	Post-test resistance measurement at Tb
Corcoef	Real	——	Correlation Coefficient
Intcpt	Real	Ω	Y intercept of Resistance vs. Temperature curve

Algorithm Syntax

The syntax for calling the algorithm directly is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the Introduction section.

Tcr_emr (V_cool, Lowrtmp, Upprtmp, Ntemps, Plot$,
Pins(*), Sweat,
Beta, Tb, Rob, Rfinal, Corcoef, Intcpt)

Methodology

The algorithm is a shell which calls the **TCR_3_EMR** algorithm to conduct the test. The shell maintains compatibility with existing HP BASIC test programs by removing/adding the following input/output parameters in the call from **TCR_EMR** to **TCR_3_EMR**:

Input parameters:

Soak: Set to 10 seconds, not supported in **TCR_EMR**.

Sweat: Removed, not used in **TCR_3_EMR**.

Output parameters:

Mode: Removed, not output by **TCR_EMR**.

XTCR_EMR

Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
V_cool	Real	Volts	100E-3	1E-3~2	Initial trial voltage to measure resistance without heating
Soak	Real	Sec	300	1~1000	Soak time at temperature
Lowrtmp	Integer	^oC	29	0~100	Lower limit for temperature range
Upprtmp	Integer	^oC	100	60~300	Upper limit for temperature range
Ntemps	Integer	——	4	4~60	Total number of at-temperatures measurements
Plot$	String	——	"Y"††		Plot/Save Data

††A "Y" plots the data on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files*.

Device Variables

Var Name	Var Type	Comment/Description
Sweat	Real	1: SWEAT structure 0: non-SWEAT (ASTM-like linear strip)

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Beta	Real	$1/^oC$	Linear coefficient of resistance
Tb	Real	^oC	Temperature at which Beta is defined. (same as Lowrtmp)
Rob	Real	W	Resistance measured at Lowrtmp
Rfinal	Real	W	Post-test resistance measurement at Tb
Corcoef	Real	——	Correlation Coefficient
Intcpt	Real	W	Y intercept of Resistance vs. Temperature curve

Algorithm Syntax

The syntax for calling the algorithm directly is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the Introduction section.

Xtcr_emr (V_cool, Soak, Lowrtmp, Upprtmp, Ntemps, Plot$,
Pins(*), Sweat,
Beta, Tb, Rob, Rfinal, Corcoef, Intcpt)

Methodology

The algorithm is a shell which calls the **TCR_3_EMR** algorithm to conduct the test. The shell maintains compatibility with existing HP BASIC test programs by removing/adding the following input/output parameters in the call from **XTCR_EMR** to **TCR_3_EMR**:

Input parameters:

Sweat: Removed, not used in **TCR_3_EMR.**

Output parameters:

Mode: Removed, not output by **XTCR_EMR**.

TVP_3_EMR

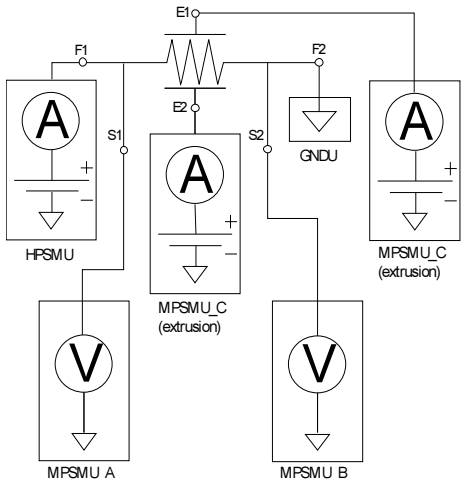
Extract the thermal resistance, Q, from temperature vs. power curve.

The **TVP_EMR** algorithm forces currents corresponding to an array of power settings through an electromigration test structure.

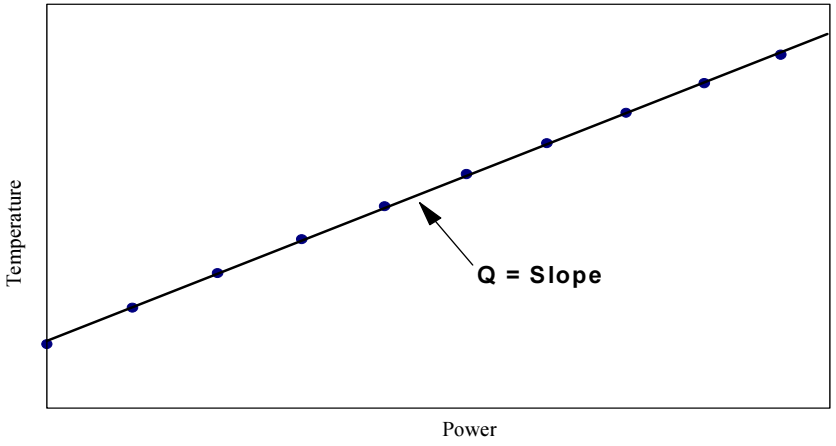
Optional extrusion monitors are supported.

Test Configuration

Note: MPSMU_C is shared during stress.



Electrical Characteristics



Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Beta†	Real	1/°C	3.55E-3	1E-3~1E-1	Linear temperature coefficient of resistance
Tb†	Real	°C	29.0	0~100	Temperature at which Beta is defined
T_peak	Real	°C	150	100~400	Forced peak temp of DUT
Chuck_tmp ††	Real	°C	29.0	20~35	Chuck temperature for this test
V_cool	Real	Volts	100E-3	1E-3~2	Initial trial voltage to measure resistance without heating
V_heat	Real	Volts	800E-3	1E-3~2	V force to produce initial heat step
Num_powr	Real	——	10	5~100	Number of power settings
Xtype	Integer	——	1	-1, 1	Execution type -1 - I, V negative with respect to GND +1 - I, V positive with respect to GND
Plot$	String	——	"Y"†††		Plot/Save Data

† Returned from TCR_3_EMR algorithm.

††Chuck_tmp is not used to control the chuck temperature but is rather the temperature the chuck is set at which can be room ambient.

†††A "Y" plots temperature versus input power on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files*.

Device Variables

Var Name	Var Type	Comment/Description
None		

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Q	Real	°C/W	Thermal resistance
Pwrint	Real	°C	Y intercept of power curve
Q_cc	Real	——	Correlation coefficient
R_o	Real	Ω	Initial resistance measurement
Ro_final	Real	Ω	Final resistance measurement after heating
T_max	Real	°C	Actual peak temperature of DUT
Mode[1]	Real	——	Failure code 00001 - short to extrusion monitor 1 00010 - short to extrusion monitor 2 00100 - structure open 00900 - could not measure Kelvin resistance 01000 - structure shorted 10000 - Del_ro > Maxro

1. Mode can return multiple failure codes

Algorithm Syntax

The syntax for calling the algorithm directly (that is, outside of Agilent SPECS) is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction* section.

Tvp_3_emr(Beta, Tb, T_peak, Chuck_tmp, V_cool, V_heat, Num_powr,
Xtype, Plot$,
Pins(*),
Q, Pwrint, Q_cc, R_o, Ro_final, T_max, Mode)

Theory of Operation

The *Introduction to Electromigration* section provides a general overview.

Q, the thermal resistance, relates the temperature rise, ΔT, (measured through resistance change) of an electromigration (EM) test structure to the power, P, dissipated by the test structure through:

$$\Delta T = QP$$

Q is used in wafer-level EM tests to estimate the current needed to reach the target temperature and to estimate damage induced changes in the initial resistance, R_o, while the test is underway. The relation between temperature rise and power is found to be linear even for geometrically complex test structures.

Note: The temperature rise referred to in this document is not the peak temperature for complex structures but the temperature calculated by resistance thermometry through the relation:

$$Temp\_now = \frac{(R - R\_o)}{\mathrm{Beta} \times \mathrm{R\_o}} + T\_amb$$

where Beta = Temperature Coefficient of Resistance (from **TCR_3_EMR**)

R = Resistance of the structure

R_o = Initial resistance at the chuck temperature

T_amb = Chuck temperature.

Ideally, Q should be measured over a temperature range that includes the temperature to be used for EM testing. The chuck is kept at a fixed temperature (Chuck_tmp) with Joule heating used to raise the structure temperature. If the maximum temperature is so large that the stripe is damaged (determined by comparing R_o measured at the beginning and end of the test), the maximum temperature should be reduced such that no damage occurs (no appreciable change in R_o). Also, instability of metal resistance (due to improperly annealed metal for example) can result in large errors in Q. This is also checked by comparing R_o at the beginning and end of the test.

At least 5 discrete applied powers spread uniformly over the range should be used in measuring Q. A good Q measurement will yield a correlation coefficient of at least 0.999.

Data File Format: When using the Plot$ variable to output ASCII data to a file the resulting format will be:

Input parameters	
Output parameters	
Power (W)	Temperature (^{o}C)
<data>	<data>

Test Structure

See "Test Structures" in the *Introduction to Electromigration* section.

Step-by-Step Action

1) Check for legal input values. If any are out of range, output variables are set to flag values 2.0E+30 and algorithm exits.
2) Calculate new Beta if Tb and Chuck_temp are different.
3) Connect SMUs as in Test Configuration diagram (Extrusion monitors are used if non-zero in device pin-out).
4) Determine forcing voltage, using V_cool as an initial guess, which will not cause heating in DUT.
5) Measure initial resistance at room temperature (R_o).The program is set for a 1 second soak time at each temperature to assure that thermal equilibrium has been reached.
6) Calculate array of powers used to heat DUT.
7) Loop through each power setting:

   a) Force current.
   b) Measure voltage drop.
   c) Calculate power; store in array.
   d) Calculate temperature in oC; store in array.

8) Measure Ro_final.
9) Find Q and Power intercept with least squares fit of temperature array vs. power array.
10) Return Q, Pwrint, Corcoef, R_o, Ro_final and T_max.

Error Code Summary

This value is assigned to the output variable if an error occurred during algorithm execution:

Value	Meaning
1.0E+30	Thermochuck has failed to reach temperature
2.0E+30	One or more input parameters are out of range
1.0E+31	DUT is bad
1.5E+30	Analysis not performed
+XE+32	Operating system error "X" occurred during execution of algorithm

Compatibility with Previous Versions

An algorithm, **TVP_EMR**, is provided to act as a parameter translation shell to maintain compatibility with HP BASIC test programs written for the obsolete **TVP_EMR**. This algorithm, which calls **TVP_3_EMR,** is briefly described below.

TVP_EMR

Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Beta†	Real	1/K	3.55E-3	1E-3~5E-3	Linear temperature coefficient of resistance
Tb†	Real	°C	29.0	0~100	Temperature at which Beta is defined
T_peak	Real	°C	150	100~400	Forced peak temp of DUT
Chuck_tmp ††	Real	°C	29.0	20~35	Chuck temperature for this test
V_cool	Real	Volts	100E-3	1E-3~2	Initial trial voltage to measure resistance without heating
V_heat	Real	Volts	800E-3	1E-3~2	V force to produce initial heat step
Num_powr	Real	——	10	5~50	Number of power settings
Plot$	String	——	"Y"†††		Plot/Save Data

† Returned from XTCR_EMR algorithm.

††Chuck_tmp is not used to control the chuck temperature but is rather the temperature the chuck is set at which can be room ambient.

†††A "Y" plots the temperature versus input power on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files*.

Device Variables

Var Name	Var Type	Comment/Description
Sweat	Real	1: SWEAT structure 0: non-SWEAT (ASTM-like linear strip)

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Q	Real	K/W	Thermal resistance
Pwrint	Real	K	Y intercept of power curve
Corcoef	Real	——	Correlation coefficient
R_o	Real	Ω	Initial resistance measurement
Ro_final	Real	Ω	Final resistance measurement after heating
T_max	Real	^{o}C	Actual peak temperature of DUT

Algorithm Syntax

The syntax for calling the algorithm directly is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the Introduction section.

Tvp_emr (Beta, Tb, T_peak, Chuck_tmp, V_cool, V_heat, Num_powr, Plot$, Pins(*), Sweat,
Q, Pwrint, Corcoef, R_o, Ro_final, T_max)

Methodology

The algorithm is a translation shell which calls the **TVP_3_EMR** algorithm to conduct the test. The shell maintains compatibility with existing HP BASIC test programs by removing/adding the following input and output parameters in the call from **TVP_EMR** to **TVP_3_EMR**:

Input parameters:

Sweat: Removed, not used in **TVP_3_EMR.**

Xtype: Set to 1, not supported in **TVP_EMR.**

Output parameters:

Mode: Removed, not output by **TVP_EMR**.

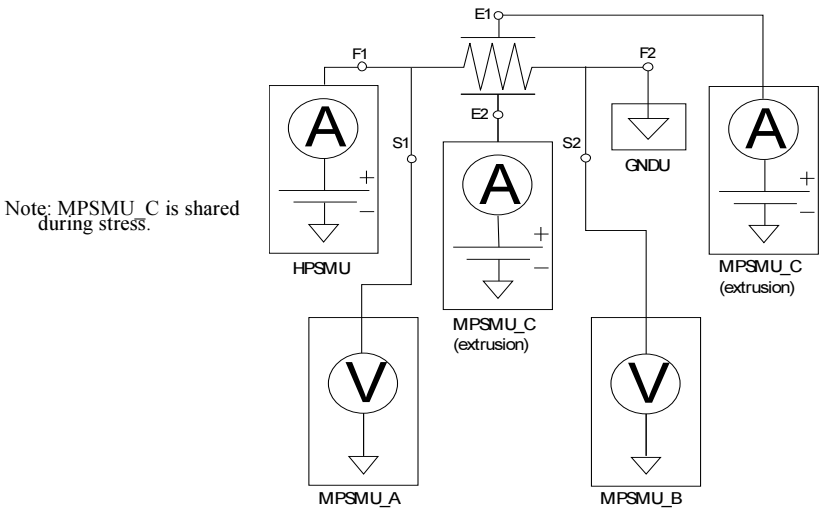
ISO_4_EMR

Isothermal "constant temperature" test to measure the time to failure of an electromigration test structure in compliance with JEDEC Standard JESD61.

The **ISO_4_EMR** algorithm maintains a constant temperature by adjusting current through an electromigration (EM) test structure until failure or time-out.

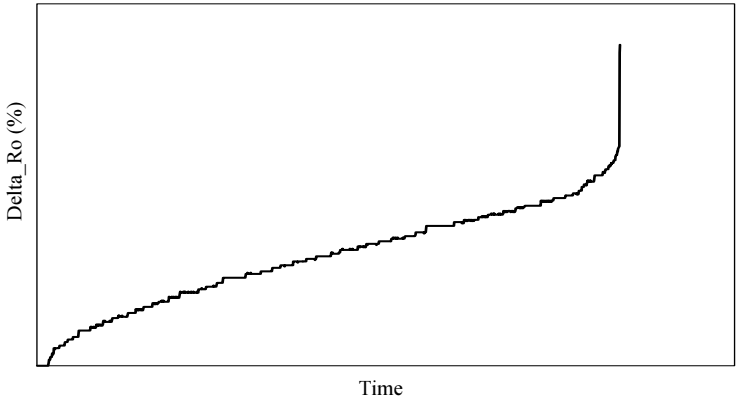
Optional extrusion monitors are supported.

Test Configuration



Note: MPSMU_C is shared during stress.

Electrical Characteristics



Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
V_cool	Real	Volts	100E-3	1E-3~2	Initial trial voltage to measure resistance without inducing heating
Target_t	Real	°C	320	150~600	Desired stress temperature
Beta[1]	Real	1/°C	3.55E-3	1E-3~1E-1	Temperature Coefficient of Resistance
Tb[1]	Real	°C	29	0~100	Temperature at which Beta is defined
Rob[1]	Real	Ω	——	1~1E3	Ro from TCR algorithm
Maxro	Real Array	%	1,5,10[2]	0.01~1E3	Maximum allowable change in Delta_ro
Mxtim	Real	Secs	1E10	1~1E36	Maximum allowable test time
Chuck_tmp[4]	Real	°C	29	——	Ambient temperature of chuck
Xtype	Integer	——	1	-2, -1, 1, 2	Execution type -2 : I, V negative, constant power mode -1 : I, V negative, constant resistance mode +1 : I, V positive, constant resistance mode +2 : I, V positive, constant power mode
Plot$	String	——	"Y"††		Plot/Save Data

1. Returned from TCR_3_EMR algorithm.

2. Defaults are the suggested values for the user to provide.

3. Described in the *Introduction to Electromigration* section.

4. Chuck_tmp is not used to control the chuck temperature but is rather the temperature the chuck is set at which can be room ambient.

††A "Y" plots Delta_ro versus time and temperature versus time on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files*.

Device Variables

Var Name	Var Type	Comment/Description
None		

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Ttf1	Real	Secs	Time to failure, first failure criterion
Ttf2	Real	Secs	Time to failure, second failure criterion
Ttf3	Real	Secs	Time to failure, third failure criterion
Tstl	Real	Secs	Time at temperature stabilization
Mode[5]	Real	——	Failure code 00001 - short to extrusion monitor 1 00010 - short to extrusion monitor 2 00100 - structure open 00200 - voltage exceeds SMU capability 00300 - current exceeds SMU capability 00900 - could not measure Kelvin resistance 01000 - structure shorted 10000 - Del_ro > Maxro 20000 - Rob/2>R_o>2*Rob
Q	Real	°C/Watt	Slope of temperature vs. power curve
Q_cc	Real	——	Correlation coefficient - Q fit
R_o	Real	Ω	Initial resistance at Chuck_temp
I_init	Real	mA	Estimated starting current
Del_ro	Real	%	Change in resistance prior to failure
Ro_final	Real	Ω	Resistance after non-destructive failure
Istl	Real	mA	Current at stabilized temperature
Ifinal	Real	mA	Current just prior to failure

5. Mode can return multiple failure codes

Algorithm Syntax

The syntax for calling the algorithm directly (that is, outside of Agilent SPECS) is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in *Introduction to Electromigration*.

Iso_4_emr(V_cool, Target_t, Beta, Tb, Rob, Maxro(*), Alpha,
Chuck_tmp, Xtype, Plot$,
Pins(*),
Ttf1, Ttf2, Ttf3, Tstl, Mode, Q, Q_cc, R_o, I_init, Del_ro,
Ro_final, Istl, Ifinal)

Theory of Operation

The *Introduction to Electromigration* section provides a general overview.

ISO_4_EMR measures the time-to-failure (TTF) of an electromigration (EM) test structure when Joule heating is used at the wafer-level to attain the target temperature. The goal is to test the structure "isothermally", that is at constant temperature. Isothermally is placed in quotes because the temperature is controlled by fixing either the structure resistance or the input power over time. The constant resistance or constant power mode is selected by the user through the input parameter Xtype.

As electromigration induced damage causes changes in the ambient temperature resistance (R_o) of the structure, control of the stripe temperature is in doubt once EM induced damage starts to occur. The chief advantage of the isothermal test over **SWEAT_3_EMR** or **CONST_I_3_EMR** is that the major independent variable, temperature, is controlled.

Methodology

The algorithm ramps the current to reach the target temperature in roughly equal temperature steps. The target current at the end of the ramp, I_start, is calculated using the initial resistance, R_o, the temperature coefficient of resistance, Beta, and the thermal resistance, Q. An accurate estimate of I_start prevents an overshoot in the structure temperature which can overstress the structure. For accurate results, it is critical to input a value of Beta

measured using the algorithm **TCR_3_EMR**. The structure thermal resistance, Q, is extracted during the initial ramp. Thus, Q is not needed as an input parameter to run the test. In fact, Q is returned as an output parameter for each structure tested. This is in contrast to the **ISO_EMR** (which is replaced by **ISO_4_EMR**), which required external measurement of Q.

The initial current ramp is useful in obtaining quantitative information on structures with defects which cause failure before the target current and temperature are reached. This is particularly important when metal lines over topography or lines containing vias are used because small variations in step coverage or via filling can cause fusing of the structure. A ramp of 20 temperature steps provides good resolution of early failures.

Once the ramp is complete, the temperature is stabilized through a control program based on **in-situ** measurement of the derivative dTEMP/dI. Once stabilized, temperature control based on a constant dTEMP/dI or dPOWER/dI is used. The value of dTEMP/dI or dPOWER/dI is calculated just prior to execution of the control loop.

Most accelerated wafer-level EM tests on narrow lines result in opens because localized severe damage is not reflected in the measured resistance. Thus, it is usually not possible to measure the change in base resistance that resulted in failure. To solve that problem, a routine is provided to estimate the percentage change in base resistance (Delta_ro) during test execution.

While wafer-level EM tests on narrow lines usually result in opens, there are occasions when that is not the case. For example, if the length of the narrow portion of the structure is "short", or large number of voids are formed in a wide line, the algorithm may be able to provide enough current control to prevent the structure from ever reaching an open condition. Thus, other failure criteria are available to terminate the test. They include limits on Delta_ro, shorts to the extrusion monitors and exceeding the maximum test time set by the user (Mxtim). If one of these non-catastrophic failure modes occurs, the program terminates the test and re-measures the base resistance, Ro_final.

In general, the time to failure is a function of the failure criterion (limit on Delta_ro) chosen. Because of that, up to three failure criteria can be set with Maxro(i) to allow such effects to be evaluated and comparisons to packaged structure tests made. The test algorithm will return a time to failure associated with each failure criterion.

Testing notes: 1) It is advisable to use an "active" chuck when performing this test. By setting the chuck to a temperature above ambient (for example 29 ^{o}C) effects due to variations in ambient temperature and chuck heating during the test are removed. 2) If the Debug exit flag is set, useful real time measurements will scroll by during the test. However, printing to the screen takes up time and if the user desires the stress current to be updated more quickly this flag can be turned off.

Data File Format: When using the Plot$ variable to output ASCII data to a file the resulting format will be:

Input parameters				
Output parameters				
Time (sec)	Current (mA)	Resistance (Ω)	Del_ro (%)	Temp (^{o}C)
<data>	<data>	<data>	<data>	<data>

Test Structure

See "Test Structures" in the *Introduction to Electromigration* section.

Step-by-Step Action

1) Check for legal input values. If any are out of range, output variables are set to flag values 2.0E+30 and algorithm exits.
2) Connect SMUs as in Test Configuration diagram (use extrusion monitors only if non-zero in pin-out).

3) Measure initial resistance using V_cool as an initial guess; record R_o.

4) Determine initial current required to heat device to target temperature.

5) Ramp current to target temperature in equal temperature steps calculating Q at each step.

6) Take initial correction step.

7) Loop to characterize temperature:

   a) Update current.

   b) Calculate temperature.

   c) Calculate change in temperature.

   d) Calculate derivative.

   e) Calculate next stress current.

   f) Exit loop if close to target temperature.

   g) Exit test if test time has exceeded maximum allowable test time (measure and record Ro_final if non-catastrophic failure).

8) Record current and power at temperature stabilization.

9) Constant slope loop:

   a) Update current.

   b) Calculate temperature and power.

   c) Calculate change in temperature or power.

   d) Calculate new stress current if outside of range set by epsilon.

   e) Exit if failure or Mxtim (measure and record Ro_final if non-catastrophic failure).

10) When exiting record Ttf(x) according to Maxro(*) criteria met; i.e. In one case if Maxro(1,2,3) were set to 1%, 5%, and 10% respectively and the structure tested had failed at a Del_ro threshold of 0.5% then all Ttf(x) would have the same time to failure. In another case if the structure could sustain a Del_ro of greater than 10% then each time-to-fail point would have different values: the time it took for Del_ro to change to 1%, 5%, and 10% respectively.

11)Return Ttf1, Ttf2, Ttf3, Tstl, Mode, Q, Q_corcoef, R_o, I_init, Del_ro, Ro_final, Istl, and Ifinal.

Error Code Summary

This value is assigned to the output variable if an error occurred during algorithm execution:

Value	Meaning
2.0E+30	One or more input parameters are out of range
1.0E+31	DUT is bad.
1.5E+30	Analysis not performed
+XE+32	Operating system error "X" occurred during execution of algorithm.

Compatibility with Previous Versions

An algorithm, **ISO_3_EMR**, is provided to act as a parameter translation shell to maintain compatibility with HP BASIC test programs written for the obsolete **ISO_3_EMR**. This algorithm, which calls **ISO_4_EMR,** is briefly described below.

Similarly, the algorithm, **XISO_EMR**, is provided to act as a parameter translation shell to maintain compatibility with HP BASIC test programs written for the obsolete **XISO_EMR**. This algorithm, which calls **ISO_3_EMR**, is breifly described following the **ISO_3_EMR** section.

ISO_3_EMR

Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
V_cool	Real	Volts	100E-3	1E-3~2	Initial trial voltage to measure resistance without inducing heating
Target_t	Real	°C	320	150~600	Desired stress temperature
Beta[1]	Real	1/°C	3.55E-3	1E-3~1E-1	Temperature Coefficient of Resistance
Tb[1]	Real	°C	29	0~100	Temperature at which Beta is defined
Rob[1]	Real	Ω	——	1~1E3	Ro from TCR algorithm
Maxro	Real Array	%	1,5,10[2]	0.01~1E3	Maximum allowable change in Delta_ro
Alpha[3]	Real	——	1	1~3	Slope of T_peak vs. T_avg
Chuck_tmp[4]	Real	°C	29	——	Ambient temperature of chuck
Xtype	Integer	——	1	-2, -1, 1, 2	Execution type -2 : I, V negative, constant power mode -1 : I, V negative, constant resistance mode +1 : I, V positive, constant resistance mode +2 : I, V positive, constant power mode
Plot$	String	——	"Y"††		Plot/Save Data

1. Returned from TCR_3_EMR algorithm.

2. Defaults are the suggested values for the user to provide.

3. Described in the *Introduction to Electromigration* section.

4. Chuck_tmp is not used to control the chuck temperature but is rather the temperature the chuck is set at which can be room ambient.

††A "Y" plots Delta_ro versus time and temperature versus time on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files*.

Device Variables

Var Name	Var Type	Comment/Description
None		

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Ttf1	Real	Secs	Time to failure, first failure criterion
Ttf2	Real	Secs	Time to failure, second failure criterion
Ttf3	Real	Secs	Time to failure, third failure criterion
Tstl	Real	Secs	Time at temperature stabilization
Mode[5]	Real	——	Failure code 00001 - short to extrusion monitor 1 00010 - short to extrusion monitor 2 00100 - structure open 00200 - voltage exceeds SMU capability 00300 - current exceeds SMU capability 00900 - could not measure Kelvin resistance 01000 - structure shorted 10000 - Del_ro > Maxro 20000 - Rob/2>R_o>2*Rob
Q	Real	°C/Watt	Slope of temperature vs. power curve
Q_cc	Real	——	Correlation coefficient - Q fit
R_o	Real	Ω	Initial resistance at Chuck_temp
I_init	Real	mA	Estimated starting current
Del_ro	Real	%	Change in resistance prior to failure
Ro_final	Real	Ω	Resistance after non-destructive failure
Istl	Real	mA	Current at stabilized temperature
Ifinal	Real	mA	Current just prior to failure

5. Mode can return multiple failure codes

Algorithm Syntax

The syntax for calling the algorithm directly (that is, outside of Agilent SPECS) is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in *Introduction to Electromigration*.

Iso_3_emr(V_cool, Target_t, Beta, Tb, Rob, Maxro(*), Alpha,
Chuck_tmp, Xtype, Plot$,
Pins(*),
Ttf1, Ttf2, Ttf3, Tstl, Mode, Q, Q_cc, R_o, I_init, Del_ro,
Ro_final, Istl, Ifinal)

Methodology

The algorithm is a translation shell which calls the **ISO_4_EMR** algorithm to conduct the test. The shell maintains compatibility with existing HP BASIC test programs by removing/adding the following input parameters in the call from **ISO_4_EMR** to **ISO_3_EMR**:

Input parameters:

Alpha^{3}: Not used in **ISO_4_EMR.**

Mxtim: Set to 1E10 by **ISO_4_EMR.**

Output parameters:

No changes.

XISO_EMR

Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
V_cool	Real	Volts	100E-3	1E-3~2	Initial trial voltage to measure resistance without inducing heating
Target_t	Real	°C	320	150~600	Desired stress temperature
Beta[1]	Real	1/°C	3.55E-3	1E-3~5E-3	Temperature Coefficient of Resistance
Tb[1]	Real	°C	29	0~100	Temperature at which Beta is defined
Rob[1]	Real	Ω	——	2~1E3	Ro from TCR algorithm
Maxro	Real Array	%	1,2,3	0.01~1E3	Maximum allowable change in Delta_ro
Tmaxq[2]	Real	°C	170	75~600	Highest temperature for Q fit
Alpha[3]	Real	——	1	1~3	Slope of T_peak vs. T_avg
Chuck_tmp[3]	Real	°C	29	20~400	Ambient temperature of chuck
Plot$	String	——	"Y"††		Plot/Save Data

1. Returned from XTCR_EMR algorithm.

2. Cannot be less than 1/2 of Target_t - i.e. user should plan ahead to take into account the maximum test temperature which will be used in a test suite.

3. Described in the *Introduction to Electromigration* section.

4. Chuck_tmp is not used to control the chuck temperature but is rather the temperature the chuck is set at which can be room ambient.

††A "Y" plots the temperature versus time and Delta_ro versus time on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files*.

Device Variables

Var Name	Var Type	Comment/Description
Sweat	Real	1: SWEAT structure 0: non-SWEAT (ASTM-like linear strip)

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Ttf1	Real	Secs	Time to failure, first point
Ttf2	Real	Secs	Time to failure, second point
Ttf3	Real	Secs	Time to failure, third point
Tstl	Real	Secs	Time at temperature stabilization
Mode[5]	Real	——	Failure code 00001 - short to extrusion monitor 1 00010 - short to extrusion monitor 2 00100 - structure open 00200 - voltage exceeds SMU capability 00300 - current exceeds SMU capability 00900 - could not measure Kelvin resistance 01000 - structure shorted 10000 - Del_ro > Maxro 20000 - Rob/2>R_o>2*Rob
Q	Real	K/Watt	Slope of temperature vs. power curve
Q_corcoef	Real	——	Correlation coefficient - Q fit
R_o	Real	Ω	Initial resistance at Chuck_temp
I_init	Real	mA	Estimated starting current
Del_ro	Real	%	Change in resistance prior to failure
Ro_final	Real	Ω	Resistance after non-destructive failure
Istl	Real	mA	Current at stabilized temperature
Ifinal	Real	mA	Current just prior to failure

5. Mode can return multiple failure codes

Algorithm Syntax

The syntax for calling the algorithm directly is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the Introduction section.

Xiso_emr(V_cool, Target_t, Beta, Tb, Rob, Maxro(*), Tmaxq, Alpha,
Chuck_tmp, Plot$,
Pins(*), Sweat,
Ttf1, Ttf2, Ttf3, Tstl, Mode, Q, Q_corcoef, R_o, I_init, Del_ro,
Ro_final, Istl, Ifinal)

Methodology

The algorithm is a translation shell which calls the **ISO_3_EMR** algorithm to conduct the test. The shell maintains compatibility with existing HP BASIC test programs by removing/adding the following input parameters in the call from **XISO_EMR** to **ISO_3_EMR**:

Input parameters:

Sweat: Removed, not used in **ISO_3_EMR.**

Xtype: Set to 1, not supported in **XISO_EMR.**

Tmaxq: Set to Target_t/2 by **ISO_3_EMR.**

Output parameters:

No changes.

BEM_3_EMR

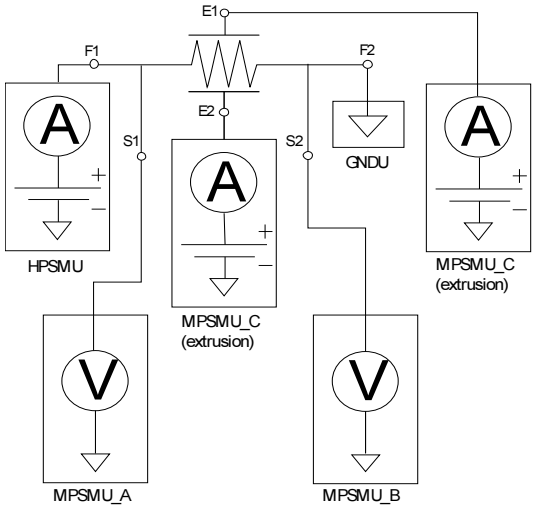
Measures the Breakdown-Energy-of-Metals in an EM test structure.

The **BEM_3_EMR** algorithm ramps the temperature in a user defined manner until failure or time-out occurs and returns the "breakdown energy".

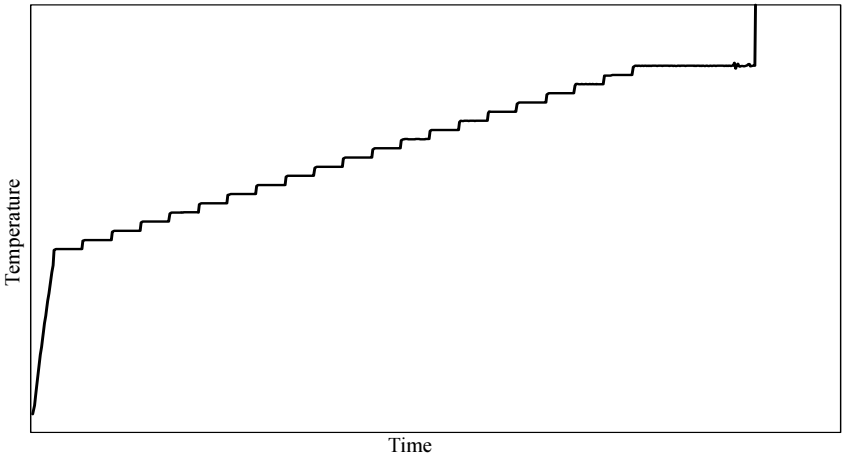
Two optional extrusion monitors are supported.

Test Configuration

Note: MPSMU_C is shared during stress.



Electrical Characteristics



Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Vcl	Real	Volts	100E-3	1E-3~2	Initial trial voltage to measure resistance without heating
Tstrt	Real	°C	200	150~600	Ramp start temperature
Tend	Real	°C	350	150~600	Ramp end temperature
Beta[1]	Real	1/°C	3.55E-3	1E-3~1E-1	Temperature Coefficient of Resistance
Tb[1]	Real	°C	29	0~100	Temperature at which Beta is defined
Rob[1]	Real	Ω	——	1~1E3	Resistance measured at Tb
Maxro	Real Array	%	1,5,10[2]	0~1000	Maximum Del_ro change points
Tmaxq[3]	Real	°C	150	100~600	Maximum temperature used to measure Q
Alpha[4]	Real	——	1	1~3	Slope of T_peak vs. T_avg
Chktmp[5]	Real	°C	29	——	Ambient temperature of chuck
Tiramp	Real	Secs	100	0~1000	Total ramp time
E_a	Real	eV	.75	0~1.2	Activation energy
L_s	Real	cm	.08	0~10	Test structure length
Plot$	String	——	"Y"††		Plot/Save Data
Nstep	Integer	——	100	1~1000	Number of temperature steps
Xtype	Integer	——	1	-1, 1	Execution type -1 : I, V negative with respect to GND +1 : I, V positive with respect to GND

1. Returned from XTCR_EMR algorithm.

2. Defaults are the suggested values for the user to provide.

3. Tmaxq should be set such that (Tstart+Chktmp)/2<Tmaxq<Tstart or else an error will be flagged.

4. Described in the *Introduction to Electromigration* section.

5. Chktmp is not used to control the chuck temperature but is rather the temperature the chuck is set at which can be room ambient.

††A "Y" plots temperature versus time and Delta_ro versus time on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files*.

Device Variables

Var Name	Var Type	Comment/Description
None		

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
E1	Real	mJ/cm	Breakdown energy, 1st pt.
E2	Real	mJ/cm	Breakdown energy, 2nd pt.
E3	Real	mJ/cm	Breakdown energy, 3rd pt.
Mode[6]	Real	——	Failure code 00001 - short to extrusion monitor 1 00010 - short to extrusion monitor 2 00100 - structure open 00900 - could not measure Kelvin resistance 01000 - structure shorted 10000 - Del_ro > Maxro 20000 - Rob/2>R_o>Rob*2
Q	Real	K/W	Slope of temp. vs. pwr. curve
Q_cc	Real	——	Correlation coefficient, Q fit
R_o	Real	Ω	Initial resistance at Chktmp
Del_ro	Real	%	Change in resistance prior to failure
Ro_fin	Real	Ω	Resistance after non-destructive failure
Istl	Real	mA	Current at stabilized temperature.
Ifin	Real	mA	Current just prior to failure
Ttf	Real	Secs	Total time to failure
Tmpf	Real	°C	Temperature at failure

6. Mode can return multiple failure codes

Algorithm Syntax

The syntax for calling the algorithm directly (that is, outside of Agilent SPECS) is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction to Electromigration* section.

Bem_3_emr(Vcl, Tstrt, Tend, Beta, Tb, Rob, Maxro(*), Tmaxq, Alpha,
Chktmp, Tiramp, E_a, L_s, Plot$, Nstep, Xtype,
Pins(*),
E1, E2, E3, Mode, Q, Q_cc, R_o, Del_ro, Ro_fin, Istl, Ifin, Ttf,
Tmpfail)

Theory of Operation

The *Introduction to Electromigration* section provides a general overview.

Algorithm **BEM_3_EMR** measures the "breakdown energy of metals", E_bd[4], of an electromigration test structure when Joule heating is used at the wafer level to attain the target temperature. The goal of the **BEM_3_EMR** test is to control the current such that the temperature is ramped in constant temperature steps. This is a departure from "classic" BEM methodologies[4], where the applied current is ramped instead. Since through Black's Equation (see *Introduction to Electromigration*), temperature is the dominant independent variable and temperature is a super-linear function of applied current, ramping temperature instead of current provides far more meaningful stresses early in the ramp. It is likely that the decision to ramp current was made because ramping the current requires no sophisticated control algorithms; a difficulty that is not an issue here.

One purported advantage of the BEM methodology is that it is a "ramp-to-failure" test that will always lead to failure within a short time, provided that the end temperature of the ramp is chosen high enough. The original work on BEM[4] rightly cautioned that the upper temperature should be limited to 350 ^{o}C as bulk diffusion starts to become important in aluminum above that temperatre. However, modern metallizations can be robust enough such that, even at 350 ^{o}C, failure will not occur rapidly enough to yield the very short test (<100 seconds) desired by BEM users.

The metric of metal reliability in the **BEM_3_EMR** test is the energy to breakdown, E_bd, defined as [1]:

where I = Applied current
R = Measured EM structure resistance (ohms)
L = EM structure length (cm)

$$E_{bd} = \int_{t_0}^{t_{fail}} \frac{I(t)^2 R(t)}{L} e^{-\frac{E_a}{kT(t)}} dt$$

[1] C.C. Hong and D.L. Crook, "Breakdown Energy of Metal (BEM) - A new technique for Monitoring Metallization Reliability at Wafer Level," Proc. 1985 International Reliability Physics Symposium, pp. 108-114, 1985.

E_a = Electromigration activation energy (eV)
k = Boltzmann's Constant (8.617×10^{-5} eV/K)
T = Temperature (K)
t = Time (s)
t_o = Test start time (s)
t_{fail} = Time to failure (s)
E_{bd} = Energy to breakdown (mJ/cm)

The energy to breakdown is normalized by test structure length in an attempt to compare the test results of different test structure types (hence the L in the equation). Also, the energy to breakdown is normalized by the acceleration factor, given by the exponential term in the equation. Because the test occurs over many different current and temperature stress conditions, this normalization is necessary to achieve consistent results from one test to another. It is absolutely critical that measured values for the activation energy be used in a **BEM_3_EMR** test.

Caveats

The **BEM_3_EMR** test has two disadvantages. The first is that because **BEM_3_EMR** is a ramped test, most of the test time is not stressful to the structure and the bulk of the damage occurs during the last step stress. This could be an advantage for metallizations with a very large defect population as the test may give better resolution of early failures. The second disadvantage is that if the test is used as a rapid "ramp-to-failure" test, the end point temperature could be chosen so high in the desire for speed that the test simply becomes a fusing test. As with all wafer-level electromigration tests, care should be taken to maintain test conditions at levels where the test is meaningful.

Methodology

The algorithm ramps the current to reach the initial ramp temperature, Tstrt, in equal temperature steps. The current at the end of the ramp, Istl, is calculated using the initial resistance R_o, the temperature coefficient of resistance, Beta, and the thermal resistance, Q. An accurate estimate of Istl prevents an overshoot in the structure temperature which can overstress the structure. It is critical to use a value of Beta measured with algorithm **TCR_3_EMR** as the input parameter. The structure thermal resistance, Q, is extracted during the initial ramp. Thus, Q is not needed as an input parameter to run the test. In fact, Q is returned as an output parameter for each structure tested.

The initial current ramp is useful in obtaining quantitative information on structures with defects which cause failure before the target current at temperature is reached. This is particularly important when structures over steps or containing vias are used because small variations in step coverage or via filling can cause fusing of the structure. A ramp of 20 steps provides good resolution of early failures.

Once the initial ramp is complete, the temperature dwells at Tstrt for Tiramp/Nstep seconds. Tiramp is the time span selected for the **BEM_3_EMR** ramp and Nstep is the number of steps chosen for the **BEM_3_EMR** ramp. At each temperature step, the temperature is controlled by the algorithms used in **ISO_3_EMR**. After the dwell time is completed, the temperature is incremented by (Tmax-Tstrt)/Nstep and the temperature dwells at Tstrt+(Tend-Tstrt)/Nstep for another Tiramp/Nstep seconds. This process continues until time Tiramp is exceeded. If the structure has survived until that point, the test continues until failure at temperature Tmax.

The breakdown energy, E_bd, is continuously integrated throughout the test.

Most wafer-level EM tests on narrow lines result in opens because localized severe damage is not reflected in the measured resistance. Thus, it is not possible to measure the change in room temperature resistance that resulted in failure because the structure is open. To solve that problem, a routine is provided to estimate the percentage change in base resistance (Delta_ro) while the test is under way.

While wafer-level EM tests on narrow lines usually result in opens, there are occasions when that is not the case. For example, if the length of the narrow portion of the structure is "short" or many voids are formed in a wide line, the algorithm may be able to provide enough current control to prevent the structure from ever reaching an open condition. Thus, other failure criteria are available to terminate the test. These include the limits on Delta_ro described above and shorts to the extrusion monitors. If these non-catastrophic failure modes occur, the program terminates the test and re-measures the base resistance, Ro_final.

Features

Because the temperature control algorithms are identical to **ISO_3_EMR** it is possible to run the **BEM_3_EMR** in an iso-thermal mode. For example, if Tstrt is set to Tend and Tiramp is set to 0, the program will stress the structure in the same manner as **ISO_3_EMR**. The output results will be three energies to breakdown, instead of three times to failure for each Maxro(i) chosen, and a single time to failure corresponding to Maxro(3). This option is useful for those who wish to stress in isothermal mode but find a breakdown energy calculation useful.

A smooth linear ramp can be selected instead of the coarsely stepped ramp usually associated with BEM testing. The smoothness of the ramp is limited by the cycle time of the algorithm, roughly 500 ms. Thus, the smoothest ramp is obtained by setting Nstep=Tiramp/0.5 where Tiramp is the ramp duration.

Testing notes: 1) It is advisable to use an "active" chuck when performing this test. By setting the chuck to a temperature above ambient (for example 29 ^{o}C) effects due to variations in ambient temperature and chuck heating during the test are removed. 2) If the Debug exit flag is set, useful real time measurements will scroll by during the test. However, printing to the screen takes up time and if the user desires the stress current to be updated more quickly this flag can be turned off.

Data File Format: When using the Plot$ variable to output ASCII data to a file the resulting format will be:

Input parameters					
Output parameters					
Time (sec)	Current (mA)	Resistance (Ω)	Del_ro (%)	Temp (^{o}C)	Energy (mJ/cm)
<data>	<data>	<data>	<data>	<data>	<data>

Test Structure

See "Test Structures" in the *Introduction to Electromigration* section.

Step-by-Step Action

1) Check for legal input values. If any are out of range, output variables are set to flag values 2.0E+30 and algorithm exits.
2) Connect SMUs as in Test Configuration diagram (use extrusion monitors only if non-zero in pin-out).
3) Measure initial resistance using Vcl as an initial guess; record R_o.
4) Determine initial current required to heat device to target temperature.
5) Ramp current to Tstrt temperature in equal temperature steps (Calculate Q in ramp using Tmaxq as last point).
6) Take initial correction step.
7) Loop to characterize temperature:
   a) Update current.
   b) Calculate temperature.
   c) Calculate change in temperature.
   d) Calculate derivative.
   e) Calculate next stress current.
   f) Exit loop if close to target temperature.
   g) Exit test if test time has exceeded maximum allowable test time (measure and record Ro_fin if non-catastrophic failure).

8) Record current at temperature stabilization.

9) Constant slope loop (Tstrt = Tend and Tiramp = 0), or step temperature loop (Tstrt < Tend and Tiramp <> 0):

   a) Integrate E_bd.

   b) Update current.

   c) Calculate temperature.

   d) Calculate change in temperature.

   e) Calculate new stress current if outside of range set by epsilon.

   f) Record breakdown energy points as per Maxro(*) input variables.

   e) Exit if failure (measure and record Ro_fin if non-catastrophic failure).

   g) Step temperature up to Tend (Tiramp/Nstep=time at temperature) if required (this will include a jump back to step 7 at the beginning of each step in order to stabilize at each temperature).

10) Return E1, E2, E3, Mode, Q, Q_cc, R_o, Del_ro, Ro_fin, Istl, Ifin, Ttf, and Tmpfail.

Error Code Summary

This value is assigned to the output variable if an error occurred during algorithm execution:

Value	Meaning
2.0E+30	One or more input parameters are out of range
1.0E+31	DUT is bad.
1.5E+30	Analysis not performed
+XE+32	Operating system error "X" occurred during execution of algorithm.

Compatibility with Previous Versions

An algorithm, **BEM_EMR**, is provided to act as a parameter translation shell to maintain compatibility with HP BASIC test programs written for the obsolete **BEM_EMR**. This algorithm, which calls **BEM_3_EMR,** is briefly described below.

BEM_EMR

Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Vcl	Real	Volts	100E-3	1E-3~2	Initial trial voltage to measure resistance without heating
Tstrt	Real	°C	200	150~600	Ramp start temperature
Tend	Real	°C	350	150~600	Ramp end temperature
Beta[1]	Real	1/°C	3.55E-3	1E-3~1E-1	Temperature Coefficient of Resistance
Tb[1]	Real	°C	29	0~100	Temperature at which Beta is defined
Rob[1]	Real	Ω	——	1~1E3	Resistance measured at Tb
Maxro	Real Array	%	1, 2, 3	0~1000	Maximum Del_ro change points
Tmaxq[2]	Real	°C	150	100~600	Highest temperature used to measure Q
Alpha[3]	Real	——	1	1~3	Slope of T_peak vs. T_avg
Chktmp[4]	Real	°C	29	——	Ambient temperature of chuck
Tiramp	Real	Secs	100	0~1000	Total ramp time
E_a	Real	eV	.75	0~1.2	Activation energy
L_s	Real	cm	.08	0~10	Test structure length
Plot$	String	——	"Y"††		Plot/Save Data
Nstep	Integer	——	100	1~1000	Number of temperature steps

1. Returned from XTCR_EMR algorithm.

2. Tmaxq should be set such that (Tstart+Chktmp)/2<Tmaxq<Tstart or else an error will be flagged.

3. Described in the *Introduction to Electromigration* section.

4. Chktmp is not used to control the chuck temperature but is rather the temperature the chuck is set at which can be room ambient.

††A "Y" plots the temperature versus time and Delta_ro versus time on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files*.

Device Variables

Var Name	Var Type	Comment/Description
Sweat	Real	1: SWEAT structure 0: non-SWEAT (ASTM-like linear strip)

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
E1	Real	mJ/cm	Breakdown energy, 1st pt.
E2	Real	mJ/cm	Breakdown energy, 2nd pt.
E3	Real	mJ/cm	Breakdown energy, 3rd pt.
Mode[5]	Real	——	Failure code 00001 - short to extrusion monitor 1 00010 - short to extrusion monitor 2 00100 - structure open 00200 - voltage exceeds SMU capability 00300 - current exceeds SMU capability 00900 - could not measure Kelvin resistance 01000 - structure shorted 10000 - Del_ro > Maxro 20000 - Rob/2>R_o>Rob*2
Q	Real	K/W	Slope of temp. vs. pwr. curve
Q_cc	Real	——	Correlation coefficient, Q fit
R_o	Real	Ω	Initial resistance at Chktmp
Del_ro	Real	%	Change in resistance prior to failure
Ro_fin	Real	Ω	Resistance after non-destructive failure
Istl	Real	mA	Current at stabilized temperature.
Ifin	Real	mA	Current just prior to failure
Ttf	Real	Secs	Total time to failure
Tmpfail	Real	°C	Temperature at failure

5. Mode can return multiple failure codes

Algorithm Syntax

The syntax for calling the algorithm directly is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction to Electromigration* section.

Bem_emr (Vcl, Tstrt, Tend, Beta, Tb, Rob, Maxro(*), Tmaxq, Alpha, Chktmp, Tiramp, E_a, L_s, Plot$, Nstep,
Pins(*), Sweat,
E1, E2, E3, Mode, Q, Q_cc, R_o, Del_ro, Ro_fin, Istl, Ifin, Ttf, Tmpfail)

Methodology

The algorithm is a translation shell which calls the **BEM_3_EMR** algorithm to conduct the test. The shell maintains compatibility with existing HP BASIC test programs by removing/adding the following input parameters in the call from **BEM_EMR** to **BEM_3_EMR**:

Input parameters:

Sweat: Removed, not used in **BEM_3_EMR.**

Xtype: Set to 1, not supported in **BEM_EMR.**

Output parameters:

No changes.

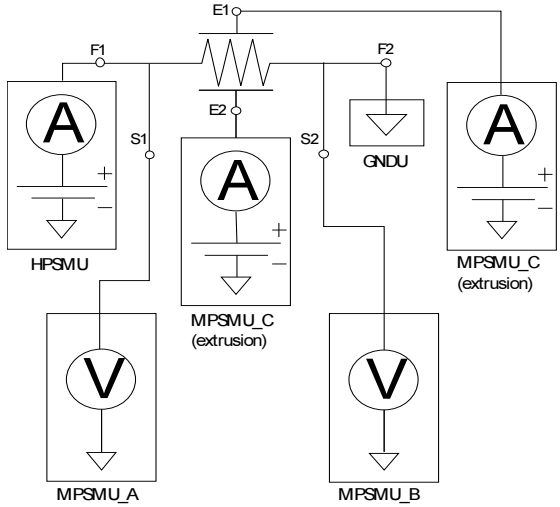
SWEAT_3_EMR

Perform "constant" time-to-failure electromigration test in compliance with JEDEC Publication JEP119.

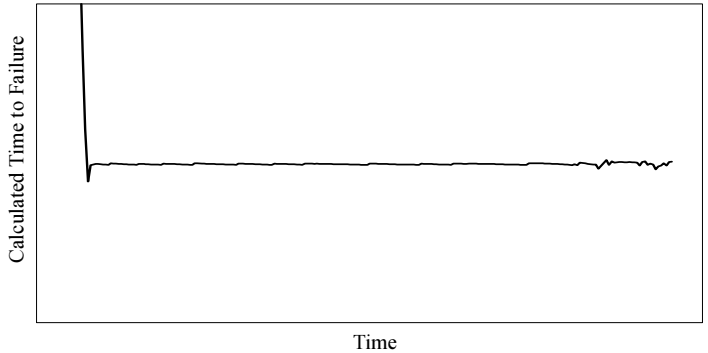
The **SWEAT_EMR** algorithm forces current through an electromigration (EM) test structure in order to maintain a constant calculated time-to-failure using Black's equation.
Optional extrusion monitors are supported.

Test Configuration

Note: MPSMU_C is shared during stress.



Electrical Characteristics



Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
V_cool	Real	Volts	100E-3	1E-3~2	Initial trial voltage to measure resistance without heating
Tart	Real	sec	1E3	1~1E6	Target time to failure
A	Real	$s\text{-}A^2/cm^4$	1E15	0~1E36	Black's equation constant
Ea	Real	eV	0.5	0~1.5	Activation energy
N	Real	——	2	0~10	Exponent for current density
Area	Real	cm^2	1E-8	1E-12~1E-6	Minimum cross-sectional area of DUT
Beta[1]	Real	$1/^oC$	3.55E-3	1E-3~1E-1	Temperature Coefficient of Resistance
Tb[1]	Real	^oC	29	0~100	Temperature at which Beta is defined
Rob[1]	Real	Ω	100	1~1E4	Resistance measured at Tb
Q[2]	Real	$^oC/W$	200	10~1E4	Slope of temperature vs. power curve
Maxro	Real	%	10	0.01~1E3	Maximum allowable change in Delta_ro
Alpha[3]	Real	——	1	1~3	Slope of peak temperature vs. average temperature
Chck_tmp[4]	Real	^oC	29	——	Ambient temperature of chuck
Plot$	String	——	"Y"††		Plot/Save Data
Xtype	Integer	——	1	-1, 1	Execution type -1 : I, V negative with respect to GND +1 : I, V positive with respect to GND

1. Returned from TCR_3_EMR algorithm.

2. Returned from TVP_3_EMR algorithm.

3. Described in the *Introduction to Electromigration* section.

4. Chuck_tmp is not used to control the chuck temperature but is rather the temperature the chuck is set at which can be room ambient.

††A "Y" plots C_ttf versus time and Delta_ro versus time on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files*.

Device Variables

Var Name	Var Type	Comment/Description
None		

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Ttf	Real	Secs	Time to failure
Tstl	Real	Secs	Time at begin of CTTF[4]
Mode[5]	Real	——	Failure code 000001 - short to extrusion monitor 1. 000010 - short to extrusion monitor 2. 000100 - structure open 000200 - voltage exceeds SMU capability 000300 - current exceeds SMU capability 00900 - could not measure Kelvin resistance 001000 - structure shorted 010000 - Del_ro > Maxro 020000 - Rob/2>R_o>2*Rob 030000 - R_stress>200*R_o
R_o	Real	Ω	Initial Resistance
A_n	Real	cm^2	Normalized area (see Theory of Operation)
Del_ro	Real	%	Change in resistance prior to failure
Ro_f	Real	Ω	Resistance after non-destructive failure
Istl	Real	mA	Current at beginning of CTTF
Ifinal	Real	mA	Current just prior to failure
Tmpstl	Real	°C	Temperature at beginning of CTTF
Tmpf	Real	°C	Temperature just prior to failure

4. CTTF - Current Time To Fail

5. Mode can return multiple failure codes.

Algorithm Syntax

The syntax for calling the algorithm directly (that is, outside of Agilent SPECS) is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction* section.

Sweat_3_emr(V_cool, Tart, A, Ea, N, Area, Beta, Tb, Rob, Q,
Maxro, Alpha, Chck_tmp, Plot$, Xtype,
Pins(*),
Ttf, Tstl, Mode, R_o, A_n, Del_ro, Ro_f, Istl, Ifinal, Tmpstl,Tmpf)

Theory of Operation

The *Introduction to Electromigration* section provides a general overview.

Algorithm **SWEAT_3_EMR** measures the time-to-failure (TTF) of an electromigration (EM) test structure when Joule heating is used at the wafer-level to attain the target temperature. The goal of the SWEAT test is to control the applied current by keeping the calculated time-to-failure (CTTF) constant. CTTF is related to the temperature, T, and applied current density, j, through Black's Equation:

$$CTTF = \frac{A}{j^{n}} e^{E_a/kT}$$

where A = Constant that depends on the material and stripe geometry ($s - \left(\frac{A}{cm^2}\right)^n$)

n = Dimensionless constant (typically 2)

E_a = Activation energy (eV)

k = Boltzmann's constant (8.617×10^{-5} eV/K)

T = Temperature in Kelvin.

j = Current density (A/cm^2)

Hints for Black's Equation

Parameters A, E_a and n are required to initiate the test. It is strongly recommended that measured values of these parameters (from package electromigration tests) be used. If not available, an initial recommendation for aluminum are the following:

Black's Term	Recommendation
E_a	0.5 eV
A	1.0×10^{12} s-$(A/cm^2)^n$
n	2

Caveat

The chief disadvantage of the **SWEAT_3_EMR** test is that no attempt is made to control either independent variable, temperature or current. Rather, an attempt is made to control the dependent variable, TTF, which can make data interpretation difficult. Since the target time-to-failure is chosen as the independent variable, the user should maintain awareness of what that target time-to-failure means in terms of temperature and current density. If the target time-to-failure is too short, temperatures can easily rise close to the melting point of the metal. In that instance there is a question as to whether the test is an electromigration test or a fusing test. To aid in interpretation of the test results, estimated temperatures and current densities are supplied as output variables.

Methodology

The algorithm ramps the current to reach the target time-to-failure in equal temperature steps. The current at the end of the ramp, I_start, is calculated using the initial resistance, R_o, the temperature coefficient of resistance, Beta, and the thermal resistance, Q. An accurate estimate of I_start prevents an overshoot in the structure temperature which can overstress the structure. It is critical that the algorithm uses a value of Q measured in algorithm **TVP_3_EMR** and a value of Beta measured from **TCR_3_EMR**.

The initial current ramp is useful in obtaining quantitative information on structures with defects which cause failure before the target current and temperature are reached. This is particularly important when metal lines over topography or containing vias are used because small variations in step coverage or via filling can cause fusing of the structure. A ramp of 20 steps provides good resolution of early failures.

Once the ramp is complete, the calculated time-to-failure (CTTF) is stabilized through a control program based on in-situ measurement of the derivative dCTTF/dI. Once stabilized, temperature control based on a constant dCTTF/dI is used. The value of dCTTF/dI is calculated just prior to execution of the constant dCTTF/dI code.

Most wafer-level EM tests on narrow lines result in opens because localized severe damage is not reflected in the measured resistance. Thus, it is usually not possible to measure the change in room temperature resistance that resulted in failure. To solve that problem, a routine is provided to estimate the percentage change in base resistance (Delta_ro) while the test is under way.

While wafer-level EM tests on narrow lines usually result in opens, there are occasions when that is not the case. For example, if the length of the narrow portion of the structure is "short" or a wide line contains many voids, the program may be able to provide enough current control to prevent the structure from ever reaching an open condition. Thus, other failure criteria are available to terminate the test. They include limits on Delta_ro and shorts to the extrusion monitors. If one of these non-catastrophic failure modes occurs, the program terminates the test and re-measures the base resistance, Ro_f.

Testing notes: 1) It is advisable to use an "active" chuck when performing this test. By setting the chuck to a temperature above ambient (for example 29 ^{o}C) effects due to variations in ambient temperature and chuck heating during the test are removed. 2) If the Debug exit flag is set, useful real time measurements will scroll by during the test. However, printing to the screen takes up time and if the user desires the stress current to be updated more quickly this flag can be turned off.

Data File Format: When using the Plot$ variable to output ASCII data to a file the resulting format will be:

Input parameters					
Output parameters					
Time (sec)	Current (mA)	Resistance (Ω)	Del_ro (%)	Temp (^{o}C)	Cttf (sec)
<data>	<data>	<data>	<data>	<data>	<data>

Test Structure

See "Test Structures" in the *Introduction to Electromigration* section.

Step-by-Step Action

1) Check for legal input values. If any are out of range, output variables are set to flag values 2.0E+30 and algorithm exits.
2) Connect SMUs as in Test Configuration diagram (use extrusion monitors only if non-zero in pin-out).
3) Measure initial resistance using V_cool as an initial guess; record R_o.
4) Estimate stress current.
5) Ramp current to target CTTF in equal temperature steps.
6) Take initial correction step.
7) Loop to characterize CTTF:
   a) Update current.
   b) Calculate change in resistance.
   c) Calculate temperature of the line.
   d) Calculate predicted time-to-failure.
   e) Calculate derivative.
   f) Calculate next stress current.
   g) (If elapsed time exceeds maxtime then measure resistance and

terminate).

8) Record the temperature at CTTF stabilization.

9) Calculate derivative from Black's equation to use in constant slope loop.

10) Constant slope loop:

a) Update current.

b) Measure resistance.

c) Calculate change in resistance.

d) Calculate temperature.

e) Calculate time-to-failure.

f) Calculate next stress current.

g) Exit if failure (measure and record R_final if non-catastrophic failure).

11) Return Ttf, Tstl, Mode, R_o, A_n, Del_ro, Ro_final, Istl, Ifinal, Tmpstl, and Tmpf.

Error Code Summary

This value is assigned to the output variable if an error occurred during algorithm execution:

Value	Meaning
2.0E+30	One or more input parameters are out of range
1.0E+31	DUT is bad
1.5E+30	Analysis not performed
+XE+32	Operating system error "X" occurred during execution of algorithm

Compatibility with Previous Versions

An algorithm, **SWEAT_EMR**, is provided to act as a parameter translation shell to maintain compatibility with HP BASIC test programs written for the obsolete **SWEAT_EMR**. This algorithm, which calls **SWEAT_3_EMR,** is briefly described below.

SWEAT_EMR

Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
V_cool	Real	Volts	100E-3	1E-3~2	Initial trial voltage to measure resistance without heating
Tart	Real	sec	1E3	1~1E6	Target time to failure
A	Real	$s\text{-}A^2/cm^4$	1E15	0~1E99	Black's equation constant
Ea	Real	eV	0.5	0~1.5	Activation energy
N	Real	——	2	0~10	Exponent for current density
Area	Real	cm^2	1E-8	1E-12~1E-6	Minimum cross-sectional area of DUT
Beta[1]	Real	1/K	3.55E-3	1E-3~5E-3	Temperature Coefficient of Resistance
Tb[1]	Real	°C	29	0~100	Temperature at which Beta is defined
Rob[1]	Real	Ω	——	1~1E3	Resistance measured at Tb
Q[2]	Real	K/W	200	10~3E3	Slope of temperature vs. power curve
Pwrint[2]	Real	K	300	250~600	Intercept of temperature vs. power curve
Maxro	Real	%	10	0.01~1E3	Maximum allowable change in Delta_ro
Alpha[3]	Real	——	1	1~3	Slope of peak temperature vs. average temperature
Chck_tmp[4]	Real	°C	29	——	Ambient temperature of chuck
Plot$	String	——	"Y"††		Plot/Save Data

1. Returned from XTCR_EMR algorithm.

2. Returned from TVP_EMR algorithm.

3. Described in the *Introduction to Electromigration* section.

4. Chuck_tmp is not used to control the chuck temperature but is rather the temperature the chuck is set at which can be room ambient.

††A "Y" plots C-ttf versus time and Delta_ro versus time on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files*.

Device Variables

Var Name	Var Type	Comment/Description
Sweat	Real	1: SWEAT structure 0: non-SWEAT (ASTM-like linear strip)

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Ttf	Real	Secs	Time to failure
Tstl	Real	Secs	Time at begin of CTTF[5]
Mode[6]	Real	——	Failure code 000001 - short to extrusion monitor 1. 000010 - short to extrusion monitor 2. 000100 - structure open 000200 - voltage exceeds SMU capability 000300 - current exceeds SMU capability 00900 - could not measure Kelvin resistance 001000 - structure shorted 010000 - Del_ro > Maxro 020000 - Rob/2>R_o>2*Rob 030000 - R_stress>200*R_o
R_o	Real	Ω	Initial Resistance
A_n	Real	cm^2	Normalized area (see Theory of Operation)
Del_ro	Real	%	Change in resistance prior to failure
Ro_final	Real	Ω	Resistance after non-destructive failure
Istl	Real	mA	Current at beginning of CTTF
Ifinal	Real	mA	Current just prior to failure
Tmpstl	Real	^oC	Temperature at beginning of CTTF
Tmpf	Real	^oC	Temperature just prior to failure

5. CTTF - Current Time To Fail

6. Mode can return multiple failure codes.

Algorithm Syntax

The syntax for calling the algorithm directly is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the Introduction section.

Sweat_emr (V_cool, Tart, A, Ea, N, Area, Beta, Tb, Rob, Q,
Pwrint, Maxro, Alpha, Chck_tmp, Plot$,
Pins(*), Sweat,
Ttf, Tstl, Mode, R_o, A_n, Del_ro, Ro_final, Istl, Ifinal,Tmpstl,
Tmpf)

Methodology

The algorithm is a translation shell which calls the **SWEAT_3_EMR** algorithm to conduct the test. The shell maintains compatibility with existing HP BASIC test programs by removing/adding the following input parameters in the call from **SWEAT_EMR** to **SWEAT_3_EMR**:

Input parameters:

Sweat: Removed, not used in **SWEAT_3_EMR.**

Xtype: Set to 1, not supported in **SWEAT_EMR.**

Pwrint: Removed, not used in **SWEAT_3_EMR.**

Output parameters:

No changes.

CONST_I_3_EMR

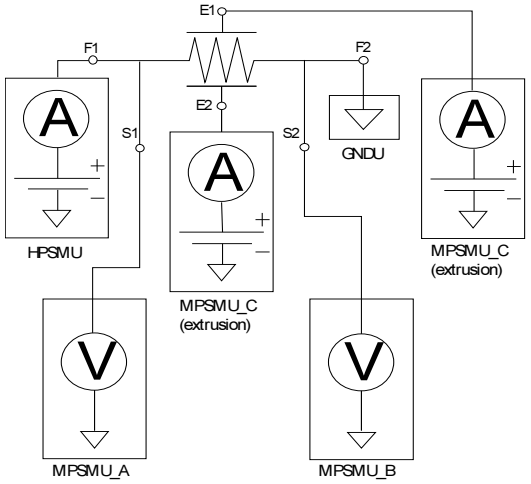
Maintain a constant current or current density through an electromigration (EM) test structure, until failure or time-out.

The **CONST_I_3_EMR** algorithm measures the time-to-failure of an electromigration (EM) test structure when a constant current or current density is applied.

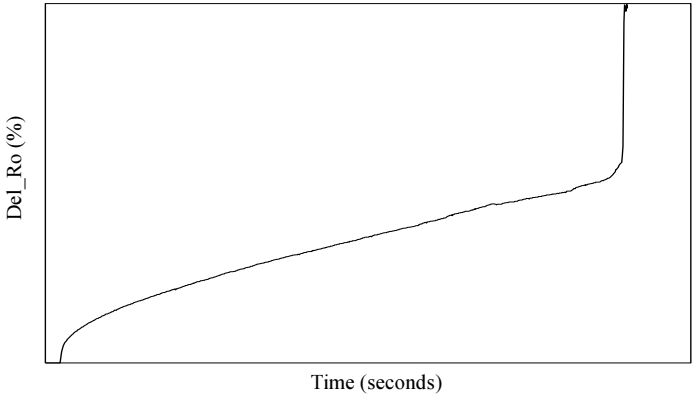
Optional extrusion monitors are supported.

Test Configuration

Note: MPSMU_C is shared during stress.



Electrical Characteristics



Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
V_cool	Real	Volts	100E-3	1E-3~2	Initial trial voltage to measure resistance without heating by using low current density.
Tar_cur	Real	mA	100	0~1E3	Target stress current 0.0 if Target_j specified.
Target_j	Real	A/cm^2	0	0~1E36	Target current density 0.0 if Tar_cur specified.
Beta[1]	Real	$1/^oC$	3.55E-3	1E-3~1E-1	Temperature Coefficient of Resistance
Tb[1]	Real	^oC	29	0~100	Temperature at which Beta is defined.
Rob[1]	Real	Ω	100	1~1E4	Resistance measured at Tb
Q[2]	Real	^oC/Watt	200	10~1E4	Slope of temperature vs. power curve
Maxro	Real	%	10	0.01~1E3	Maximum allowable change in Delta_ro
Alpha[3]	Real	——	1	1~3	Slope of T_peak vs. T_avg
Chuck_temp[4]	Real	^oC	29	20~400	Temperature of the chuck
Area	Real	cm^2	1e-12	1E-12~1E-6	Device cross-sectional area
Xtype	Integer	——	1	-1, 1	Execution type -1 - I, V negative with respect to GND +1 - I, V positive with respect to GND
Plot$	String	——	"Y"††		Plot/Save Data

1. Returned from TCR_3_EMR algorithm.

2. Returned from TVP_3_EMR algorithm.

3. Described in the *Introduction to Electromigration* section.

4. Chuck_tmp is not used to control the chuck temperature but is rather the temperature the chuck is set at which can be room ambient.

††A "Y" plots Delta_ro verus time. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files.*

Device Variables

Var Name	Var Type	Comment/Description
None		

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Ttf	Real	Sec	Time to failure
Tistl	Real	Sec	Time at end of current ramp
Mode[5]	Real	——	Failure code 00001 - short to extrusion monitor 1. 00010 - short to extrusion monitor 2 00100 - structure open 00200 - voltage exceeds SMU capability 00300 - current exceeds SMU capability 00900 - could not measure Kelvin resistance 01000 - structure shorted 10000 - Del_ro > Maxro 20000 - Rob/2>R_o>2*Rob
Del_ro	Real	%	Change in resistance prior to failure
R_o	Real	Ω	Initial resistance at Chuck_temp
Ro_final	Real	Ω	Resistance after non-destructive failure
Tstl	Real	^{o}C	Temperature at end of ramp.
Tfinal	Real	^{o}C	Temperature just prior to failure.

5. Mode can return multiple failure codes

Algorithm Syntax

The syntax for calling the algorithm directly (that is, outside of Agilent SPECS) is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction* section.

Const_i_3_emr(V_cool, Tar_cur, Target_j, Beta, Tb, Rob, Q,
Maxro, Alpha, Chuck_temp, Area, Xtype, Plot$,
Pins(*),
Ttf, Tistl, Mode, Del_ro, R_o, Ro_final, Tstl, Tfinal)

Theory of Operation

The *Introduction to Electromigration* section provides a general overview.

Algorithm **CONST_I_3_EMR** measures the time-to-failure (TTF) of an electromigration (EM) test structure where a constant current (or current density) is used at the wafer-level to stress the structure. The chief feature of the constant current test is that the minor independent variable, current (or current density), is controlled.

Methodology

The algorithm ramps the current to reach the target current in roughly equal temperature steps. The temperature at the end of the ramp, Tstl, is calculated using the initial resistance, R_o, and the temperature coefficient of resistance, Beta. It is critical the algorithm uses a value of Beta measured from **TCR_3_EMR**.

The initial current ramp is useful in obtaining quantitative information on structures with defects which cause failure before the target current is reached. This is particularly important when metal lines containing vias are used because small variations in via filling can cause fusing of the structure. A ramp of 20 steps provides good resolution of early failures. Once the ramp is complete, the current is fixed at the target level.

Most accelerated wafer-level EM tests on narrow lines result in opens because localized severe damage is not reflected in the measured resistance. Thus, it is usually not possible to measure the change in room temperature

resistance that resulted in failure. To solve that problem, a routine is provided to estimate the percentage change in base resistance (Delta_ro) while the test is under way.

While wafer-level EM tests on narrow lines usually result in opens, there are occasions when that is not the case such as multi-void formation in wide lines. Thus, other failure criteria are available to terminate the test. They include limits on Delta_ro and shorts to the extrusion monitors. If one of these non-catastrophic failure modes occurs, the program terminates the test and re-measures the base resistance, Ro_final.

Testing notes: 1) It is advisable to use an "active" chuck when performing this test. By setting the chuck to a temperature above ambient (for example 29^{o} C) effects due to variations in ambient temperature and chuck heating during the test are removed. 2) If the Debug exit flag is set, useful real time measurements will scroll by during the test. However printing to the screen takes up time and if the user desires the resistance to be updated more quickly this flag can be turned off.

Data File Format: When using the Plot$ variable to output ASCII data to a file the resulting format will be:

Input parameters				
Output parameters				
Time (sec)	Current (mA)	Resistance (Ω)	Del_ro (%)	Temp (^{o}C)
<data>	<data>	<data>	<data>	<data>

Test Structure

See "Test Structures" in the *Introduction to Electromigration* section.

Step-by-Step Action

1) Check for legal input values. If any are out of range, output variables are set to flag values 2.0E+30 and algorithm exits.
2) Connect SMUs as in Test Configuration diagram (use extrusion monitors only if non-zero in pin-out).
3) Measure initial resistance using V_cool as an initial guess; record R_o.
4) Determine initial current required to achieve target current density (if selected).
5) Ramp current to target current in equal temperature steps.
6) Record temperature at end of ramp.
7) Constant current loop.
   a) Calculate temperature.
   b) Exit if failure (measure and record Ro_final if non-catastrophic failure).
8) Return Ttf, Tistl, Mode, Del_ro, R_o, Ro_final, Tstl, and Tfinal.

Error Code Summary

This value is assigned to the output variable if an error occurred during algorithm execution:

Value	Meaning
2.0E+30	One or more input parameters are out of range
1.0E+31	DUT is bad
1.5E+30	Analysis not performed
+XE+32	Operating system error "X" occurred during execution of algorithm

Compatibility with Previous Versions

An algorithm, **CONST_I_EMR**, is provided to act as a parameter translation shell to maintain compatibility with HP BASIC test programs written for the obsolete **CONST_I_EMR**. This algorithm, which calls **CONST_I_3_EMR**, is briefly described below.

CONST_I_EMR

Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
V_cool	Real	Volts	100E-3	1E-3~2	Initial trial voltage to measure resistance without heating by using low current density.
Tar_cur	Real	mA	100	0~1E3	Target stress current 0.0 if Target_j specified.
Target_j	Real	A/cm^2	0	-1E99~1E99	Target current density 0.0 if Tar_cur specified.
Beta[1]	Real	$1/^oC$	3.55E-3	1E-3~5E-3	Temperature Coefficient of Resistance
Tb[1]	Real	^oC	29	0~100	Temperature at which Beta is defined.
Rob[1]	Real	Ω	——	2~1E3	Resistance measured at Tb
Q[2]	Real	K/Watt	200	10~3000	Slope of temperature vs. power curve
Pwrint[2]	Real	K	300	250~600	Intercept of temperature vs. power curve
Maxro	Real	%	100	0.01~1E3	Maximum allowable change in Delta_ro
Alpha[3]	Real	——	1	1~3	Slope of T_peak vs. T_avg
Chuck_temp[4]	Real	^oC	29	20~400	Temperature of the chuck
Area	Real	cm^2	1E-12	1E-12~1E-6	Device cross-sectional area
Plot$	String	——	"Y"††		Plot/Save Data

1. Returned from TCR_3_EMR algorithm.

2. Returned from TVP_3_EMR algorithm.

3. Described in the *Introduction to Electromigration* section.

4. Chuck_tmp is not used to control the chuck temperature but is rather the temperature the chuck is set at which can be room ambient.

††A "Y" plots Delta_ro verus time. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files*.

Device Variables

Var Name	Var Type	Comment/Description
Sweat	Real	1: SWEAT structure 0: non-SWEAT (ASTM-like linear strip)

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Ttf	Real	Sec	Time to failure
Tistl	Real	Sec	Time at end of current ramp
Mode[5]	Real	——	Failure code 00001 - short to extrusion monitor 1. 00010 - short to extrusion monitor 2 00100 - structure open 00900 - could not measure Kelvin resistance 01000 - structure shorted 10000 - Del_ro > Maxro 20000 - Rob/2>R_o>2*Rob
Del_ro	Real	%	Change in resistance prior to failure
R_o	Real	Ω	Initial resistance at Chuck_temp
Ro_final	Real	Ω	Resistance after non-destructive failure
Tstl	Real	°C	Temperature at end of ramp.
Tfinal	Real	°C	Temperature just prior to failure.

5. Mode can return multiple failure codes

Algorithm Syntax

The syntax for calling the algorithm directly is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the Introduction section.

Const_i_emr (V_cool, Tar_cur, Target_j, Beta, Tb, Rob, Q,
Pwrint, Maxro, Alpha, Chuck_temp, Area, Plot$,
Pins(*), Sweat,
Ttf, Tistl, Mode, Del_ro, R_o, Ro_final, Tstl, Tfinal)

Methodology

The algorithm is a shell which calls the **CONST_I_3_EMR** algorithm to conduct the test. The shell maintains compatibility with existing HP BASIC

ISO_EMR

Isothermal test to measure the time to failure.

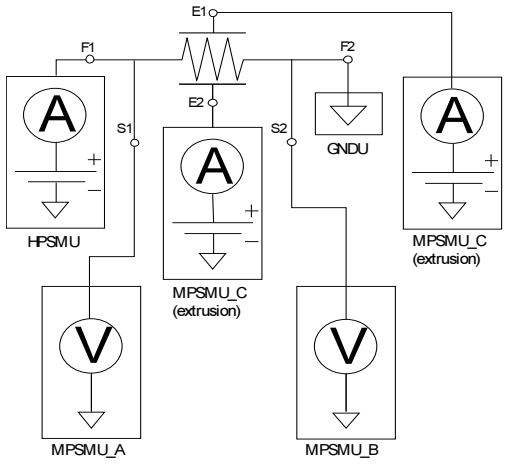
The **ISO_EMR** algorithm maintains a constant temperature by adjusting current through an electromigration (EM) test structure until failure or time-out.

Optional extrusion monitors are supported.

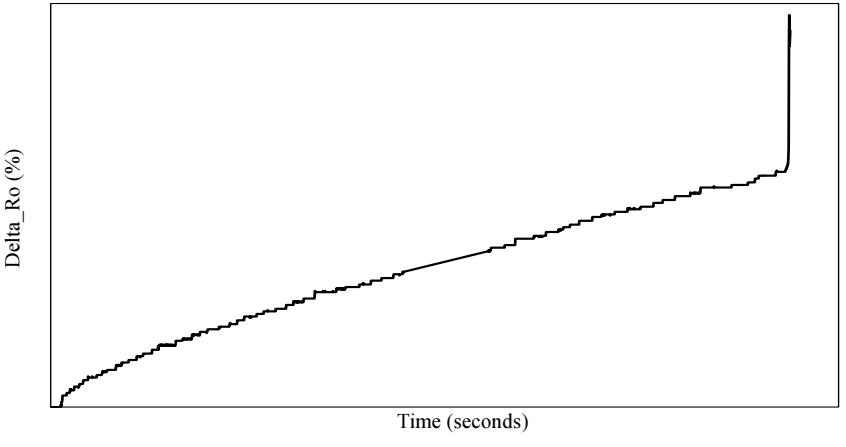
Note: **ISO_EMR** is obsolete and replaced by **ISO_4_EMR**. This documentation is for migration and compatibility purposes.

Test Configuration

Note: MPSMU_C is shared during stress.



Electrical Characteristics



Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
V_cool	Real	Volts	100E-3	1E-3~2	Initial trial voltage to measure resistance without heating
Target_t	Real	°C	320	0~600	Desired stress temperature
Beta[1]	Real	1/K	3.55E-3	1E-3~5E-3	Temperature Coefficient of Resistance
Tb[1]	Real	°C	29	0~100	Temperature at which Beta is defined
Rob[1]	Real	Ω	——	2~1E3	Resistance measured at Tb
Q[2]	Real	K/Watt	200	10~3E3	Slope of temperature vs. power curve
Pwrint[2]	Real	K	300	250~600	Intercept of temperature vs. power curve
Maxro	Real	%	100	0.01~1E3	Maximum allowable change in Delta_ro
Alpha[3]	Real	——	1	1~3	Slope of T_peak vs. T_avg
Chuck_tmp[4]	Real	°C	29	20~400	Temperature of the chuck
Plot$	String	——	"Y"††		Plot/Save Data

1. Returned from TCR_3_EMR algorithm.

2. Returned from TVP_3_EMR algorithm.

3. Described in the *Introduction to Electromigration* section.

4. Chuck_tmp is not used to control the chuck temperature but is rather the temperature the chuck is set at which can be room ambient.

††A "Y" plots Delta_ro verus time and temperature verus time. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files*.

Device Variables

Var Name	Var Type	Comment/Description
Sweat	Real	1: SWEAT structure 0: non-SWEAT (ASTM-like linear strip)

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Ttf	Real	Secs	Time to failure
Tstl	Real	Secs	Time at temperature stabilization.
Mode[5]	Real	——	Failure code 00001 - short to extrusion monitor 1. 00010 - short to extrusion monitor 2. 00100 - structure open 00200 - voltage exceeds SMU capability 00300 - current exceeds SMU capability 00900 - could not measure Kelvin resistance 01000 - structure shorted 10000 - Del_ro > Maxro
R_o	Real	Ω	Initial resistance at Chuck_temp
I_init	Real	mA	Estimated starting current.
Del_ro	Real	%	Change in resistance prior to failure
Ro_final	Real	Ω	Resistance after non-destructive failure
Istl	Real	mA	Current at stabilized temperature.
Ifinal	Real	mA	Current just prior to failure

5. Mode can return multiple failure codes

Algorithm Syntax

The syntax for calling the algorithm directly is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction* section.

```
Iso_emr  (V_cool, Target_t, Beta, Tb, Rob, Q, Pwrint, Maxro, Alpha,
          Chuck_tmp, Plot$
          Pins(*), Sweat,
          Ttf, Tstl, Mode, R_o, I_init, Del_ro, Ro_final, Istl, Ifinal)
```

Theory of Operation

See **ISO_4_EMR** for theory of operation. Differences between **ISO_EMR** and **ISO_4_EMR** are:

ISO_EMR supports a single definition of failure while **ISO_4_EMR** supports three definitions of failure.

ISO_EMR requires input of the thermal resistance, Q, measured using **TVP_3_EMR** while **ISO_4_EMR** performs this measurement in-situ and outputs Q.

Data File Format: When using the Plot$ variable to output ASCII data to a file the resulting format will be:

Input parameters				
Output parameters				
Time (sec)	Current (mA)	Resistance (Ω)	Del_ro (%)	Temp (^{o}C)
<data>	<data>	<data>	<data>	<data>

Test Structure

See "Test Structures" in the *Introduction to Electromigration* section.

Step-by-Step Action

1) Check for legal input values. If any are out of range, output variables are set to flag values 2.0E+30 and algorithm exits.
2) Connect SMUs as in Test Configuration diagram (use extrusion monitors only if non-zero in pin-out).
3) Measure initial resistance using V_cool as an initial guess; record R_o.
4) Determine initial current required to heat device to target temperature.
5) Ramp current to target temperature in equal temperature steps.
6) Take initial correction step.
7) Loop to characterize temperature:
   a) Update current.
   b) Calculate temperature.
   c) Calculate change in temperature.
   d) Calculate derivative.
   e) Calculate next stress current.
   f) Exit loop if close to target temperature.
   g) Exit test if test time has exceeded maximum allowable test time (measure and record Ro_final if non-catastrophic failure).
8) Record current at temperature stabilization.
9) Constant slope loop:
   a) Update current.
   b) Calculate temperature.
   c) Calculate change in temperature.
   d) Calculate new stress current if outside of range set by epsilon.
   e) Exit if failure (measure and record Ro_final if non-catastrophic failure).
10) Return Ttf, Tstl, Mode, R_o, I_init, Del_ro, Ro_final, Istl, and Ifinal.

Error Code Summary

This value is assigned to the output variable if an error occurred during algorithm execution:

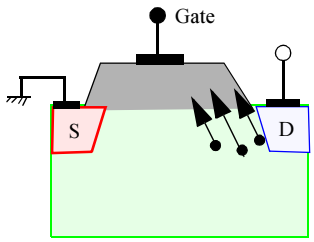
Value	Meaning
2.0E+30	One or more input parameters are out of range
1.0E+31	DUT is bad
1.5E+30	Analysis not performed
+XE+32	Operating system error "X" occurred during execution of algorithm

6 Hot Carriers and Device Reliability

Introduction to Hot Carriers (HC)

This section provides a general overview of the WLR hot-carrier-injection (HCI) and device reliability algorithms.

Small channel length in MOSFET leads to high electric fields localized close to drain terminal



High electric fields cause energetic carriers — "Hot Carriers"

Hot Carriers create interface states or become trapped in oxide

Vth, Gm, Ids, etc. degrade due to interface-states or trapped charge

Hot Carrier and Device Reliability Algorithms†

Name	Device Type	Comment/Description
MAXSTRS	MOSFET	Estimated gate voltage for maximum hot carrier damage
HCI_3	MOSFET	Complete non-programming hot carrier tests - recommended
STRSMEAS_2	MOSFET	Constant voltage stress, measures Id, Ib, Ig
HCI	MOSFET	Obsolete generic interface to user-written custom hot carrier test
HCSTRESS	MOSFET	Constant voltage hot carrier stress
CPV_3	MOSFET	Charge pumping (by voltage)
VT	MOSFET	JEDEC (JESD28 & JESD60) compliant threshold voltage (Vt), transconductance, (Gm), and drain current (Id)
CALC_VT	MOSFET	Calculate change in Vt, Gm, and Id
LIFETIME_2	MOSFET	Calculate "Lifetime" in compliance with JEDEC JESD28 & JESD60
IDIBIG	MOSFET	Drain, bulk, and gate currents and user-specified voltages
CALC_ID	MOSFET	Calculate change in currents
CPF_3	MOSFET	ASTM F1340-91 compliant charge pumping (by frequency) for average interface-state density
IDVD	MOSFET	Id vs. Vd characteristics and output conductance, Gd
S	MOSFET	Subthreshold slope
VPT	MOSFET	Punchthrough voltage
DELTAL	M2FET	Parasitic source and drain resistance and Delta L
DOP	MOSFET	Average substrate doping
HC_LINK_2	MOSFET	Automatic link between lifetime and measurements. Used by HCI_3 for hot carrier tests. May be useful plasma damage.
HC_LINK	MOSFET	Obsolete link algorithm
PD_LINK	MOSFET	Automatic link between plasma damage and structure type.

†Using HCI_3 gives a JEDEC compliant hot carrier test (JESD28 or JESD60).

Test Structure Definition

The Hot Carrier and Device Reliability algorithms are defined for the device type MOSFET and M2FET. This structure can also be used for plasma damage.

MOSFET Device Specifications

Terminal Connections	Position in Pins Array	Comment/Description
Gate	1	Gate terminal
Drain	2	Drain terminal
Source	3	Source terminal
Bulk	4	Bulk (or well) terminal
Sub	5	Substrate contact†
Chuck	6	Wafer chuck†

Var Name	Var Type	Comment/Description
Width	Real	um, Mask channel width
Length	Real	um, Mask channel length
Type$	String	(N/P), Channel type

† Not used by HC and device reliability algorithms

M2FET Device Specifications†

Terminal Connections	Position in Pins Array	Comment/Description
Gate1	1	Gate terminal of transistor 1
Drain1	2	Drain terminal of transistor 1
Src1	3	Source terminal of transistor 1
Bulk1	4	Bulk terminal of transistor 1
Gate2	5	Gate terminal of transistor 2
Drain2	6	Drain terminal of transistor 2
Src2	7	Source terminal of transistor 2
Bulk2	8	Bulk terminal of transistor 2
Sub	9	Substrate contact
Chuck	10	Wafer chuck

Var Name	Var Type	Comment/Description
Width1	Real	um, Mask channel width of transistor 1
Length1	Real	um, Mask channel length of transistor 1
Width2	Real	um, Mask channel width of transistor 2
Length2	Real	um, Mask channel length of transistor 2
Type$	String	(N/P), Channel Type

† Used by DELTAL_M2F algorithm.

Theory of Operation

For MOSFET devices, small channel lengths lead to high electric fields in the drain region. These high electric fields cause energetic carriers — Hot Carriers — which create interface states or become trapped in the oxide. The damage to the SiO_2 or SiO_2 - Si interface is reflected in the shifts of key transistor parameters such as threshold voltage or maximum transconductance. According to JEDEC standards JESD28 and JESD60, accelerated hot-carrier-induced degradation of MOS transistors is performed by repeating a stress/measure sequence: applying a stress voltage at the gate, drain and bulk terminals for a specified time and then measuring the change in parameter(s) (Figure 1). Typically, the stress is repeated until a pre-defined total stress time has expired.

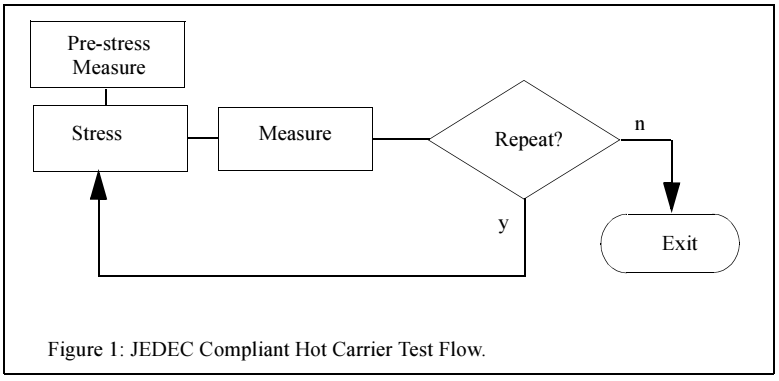


Figure 1: JEDEC Compliant Hot Carrier Test Flow.

The **LIFETIME_2_MOS** algorithm is used to extrapolate (or interpolate) the data to a desired change to determine the lifetime under these stress conditions. These parameters can include, but are not limited to, drain current, maximum transconductance, threshold voltage, charge pumping current and voltage, output conductance, subthreshold slope and punchthrough voltage.

To complement the stress/measurement algorithms, the following algorithms provide insight into the physics of the device and allow comparison between devices:

- Estimated Gate Voltage for Max Hot Carrier Damage (**MAXSTRS_MOS**)
- Source/Drain Resistance and Effective Channel Length (**DELTAL_M2F**)
- Substrate Doping (**DOP_MOS**)

Hot Carrier Test Methods

Creating hot-carrier tests outside of the Agilent SPECS test shell is very quick and easy, however creating a test within the test shell can require some planning. There are two methods of implementing a hot carrier test:

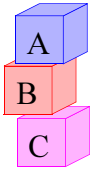
1) Sequence Method (suitable for production tests)

2) **HCI_3** Method (suitable for development and qualification tests)

Sequence Method

In the sequence method, an Agilent SPECS Test Plan is constructed from the hot carrier algorithms which function as hot carrier building blocks. This method is designed for quickly developing short, simple tests for production.

Example Hot Carrier Building Blocks



- Apply Constant Voltage Stress
  - STRSMEAS_2_MOS (recommended)
- Measure parameter(s) of interest
  - VT_MOS for Vt, Gm, & Id
  - IDIBIG_MOS for Drain, Bulk, & Gate Currents
- Calculate change due to stress
  - CALC_VT_MOS for VT_MOS
  - CALC_ID_MOS for IDIBIG_MOS
- Predict time of target degradation
  - LIFETIME_2_MOS

Figure 2: Example Building Blocks.

Figure 2 describes the basic "building block" algorithms for a hot carrier test using the sequence method. The application of a voltage stress followed by one or more measurements forms a stress/measure cycle. Several of these cycles are required to characterize a device's suseptibility to hot carriers.

Because we are interested in the change due to stress, a "calc" algorithm calculates the change in a parameter(s) due to stress. Typically, after a series of stress/measure cycles, the **LIFETIME_2_MOS** algorithm is used to predict a target degradation such as 10%. In the Agilent SPECS test shell, which can perform looping, logical operation and calculations, the sequence method is easy and powerful. This simplifies development of hot carrier tests.

Figure 3 shows an example Agilent SPECS Test Plan for a hot carrier test using the sequential algorithm approach.

Sample Agilent SPECS Sequence Method

Algorithm	Input	Output
LET	Times[1]=0	
LET	Times[2]=0.1	
LET	Times[3]=0.3	
LET	Times[4]=0.7	
LET	Times[5]=1	
LET	Times[6]=3	
LET	Ntimes=5	
LET	I=1	
VT_MOS_XL		Vtext_0,Vtci_0,Gm_0,Idsat_0
WHILE	(I <= Ntimes)	
STRSMEAS_2_MOS_XL	Time=Times[I+1],Acc_tm=Times[I]	Tstress
VT_MOS_XL		Vtext_1,Vtci_1,Gm_1,Idsat_1
CALC_VT_MOS_XL	Id=Idsat_0,Id_p=Idsat_1	,,,,,,,PIdsat
LET	Pa[I]=PIdsat	
LET	Times[I]=Tstress	
LET	I=I+1	
END WHILE		
LIFETIME_2_MOS_XL	Pname="Idsat",Pa=Pa,Ti=Times,Change=5,Npts=Ntimes,Lpts=Ntimes	Tau_Idsat,,Slope

Pre-stress

Stress/Meas

Repeat S/M

Figure 3: Example Agilent SPECS Hot Carrier Testplan Suitable for Production.

HCI_3 Algorithm Method

Although the sequence method is a fast and easy way to implement simple hot-carrier tests, it can be cumbersome for longer or more complex tests that might be run in development or qualification. In these cases, the **HCI_3_MOS** algorithm simplifies the test and avoids a long algorithm sequence or programming. **HCI_3_MOS** conducts a user specified hot carrier test using **HCI_3_MOS** builder which is easy to use. The user specifies all the stress times and measurement routines in an easy-to-use graphical interface. No programming is needed. We have also provided example tests for various technologies which the user can start using with little modification. See **HCI_3_MOS** algorithm for more information.

Test Structure Design Rules

The hot carrier algorithms require an N-channel or P-channel MOSFET to perform hot-carrier-induced degradation. At a minimum, the device should include separate gate and drain contacts, but ideally would include the following:

- Separate gate, source, drain and substrate terminals which would permit independent control over the gate voltage and drain voltage as well as independent monitoring of the substrate and gate currents.
- A large substrate contact encircling the device. This reduces the source to substrate resistance which increases the maximum drain stress voltage. A larger drain voltage reduces test time.

Applicable Test Structures

For users of PDQ-FAB WLR test structures, a separate product available from Agilent, the following table lists PDQ-WLR software algorithms recommended for use with the test structure.

PDQ-FAB WLR Hot Carrier and Device Reliability Test Structures

Structure	Structure Name	Primary Applicable WLR Algorithms
Enhanced Hot Carrier structures for faster predictive HC tests	HCXTR_N HCXTR_P HCXTRV_N HCXTRV_P HCXTROT	HCI _3 (Recommended) STRSMEAS_2 VT CPV_3 IBIDIG IDVD CPF_3 S VPT MAXSTRS

The next table lists some of the possible failure mechanisms detected using the PDQ-WLR test structures and software.

Hot Carrier and Device Reliability Test Structures and Failure Mechanisms

Test Structure Description	Failure Mechanisms	Tests
Fast Hot Carrier	Implant Shadowing; TEOS and SOG Water Outgassing; S/D Anneal & Implant; LDD Overlap Spacing; Interface Uniformity & Charge Trapping; Insufficient Interface Anneal; Hydrogen Contamination; Channel Length Variation	Hot Carrier & Charge Pumping

Plasma Damage

This section provides a quick overview of the plasma-damage detection using the device reliability algorithms. Please refer to *Introduction to Hot Carriers* section for more information on algorithms and device specifications.

Process-induced plasma-damage occurs in many fabrication sequences such as plasma etching and photoresist stripping. Plasma damage can cause device parametric shift or oxide breakdown. Using special test structures, this plasma damage can be detected. These test structures collect the plasma charging current with various conducting antennas. The charge is routed to a device where it causes damage to the gate oxide and Si-SiO_2 interface.

The **HCSTRESS_MOS** algorithm can be used along with **VT_MOS** and/or **CPV_3_MOS** to determine the impact of process-induced plasma damage on device reliability. For process monitoring, plasma damage test structure(s) are first measured with **VT_MOS** and/or **CPV_3_MOS** algorithms. Then, a short hot carrier stress is performed to activate latent plasma damage. A second **VT_MOS** and/or **CPV_3_MOS** is measured. The change in threshold voltage, transconductance or interface state density is measured. This change is then compared to a reference test structure (no plasma damage) exposed to the same hot carrier stress. A simple example test in Agilent SPECS is in the following figure. Here the threshold voltage of a reference and a potentially plasma-damaged device is measured before and after a hot carrier stress. In this example, the threshold voltage difference between the reference device and the plasma device both before and after stress is used to measure plasma-damage. In many cases, the post-stress threshold voltage difference will be a more sensitive indicator of plasma damage. The **PD_LINK_MOS** algorithm stores this information in the database for easy retrieval. Further information about **PD_LINK_MOS** operation may be found in the *PD_LINK_MOS* section.

In addition to a hot-carrier stress, a non-destructive oxide stress can be used with plasma damage structures. **JTDDB_3_CAP** can be used along with the **VT_MOS** and/or **CPV_3_MOS** to monitor the susceptibility of an oxide to process-induced plasma damage.

In development, one may wish to quantitatively assess the impact of plasma damage on device reliability. This can be accomplished by performing a series of hot-carrier test (using **HCI_3_MOS**) on the special plasma devices.

Sample Plasma Damage Test in Agilent SPECS

Algorithm	Input	Output
VT_MOS_XL		A,Vtref_0,B,C
HCSTRESS_MOS_XL	Time=10	
VT_MOS_XL		D,Vtref_1,E,F
VT_MOS_XL		G,Vtant_0,H,I
HCSTRESS_MOS_XL	Time=10	
VT_MOS_XL		J,Vtant_1,K,L
PD_LINK_MOS_XL	Test_name = "Prestress", Dut_type = "Poly", Ant_type = "Edge", Meas_type = "dVt", Ref_rsvp = Vtref_0, Dut_rsvp = Vtant_0, Vds, Vgs, Vbs - same as HCSTRESS_MOS inputs	
PD_LINK_MOS_XL	Test_name = "Poststress", Dut_type = "Poly", Ant_type = "Edge", Meas_type = "dVt", Ref_rsvp = Vtref_1, Dut_rsvp = Vtant_1, Vds, Vgs, Vbs - same as HCSTRESS_MOS inputs	

Antenna Device

Reference Device

Applicable Test Structures

For users of PDQ-FAB WLR test structures, a separate product available from Agilent, the following table lists PDQ-WLR software algorithm recommended for use with the test structure.

PDQ-WLR Plasma Damage Test Structures

Structure	Structure Name	Primary Applicable WLR Algorithms
Process-Induced Plasma Damage. Area, Edge, and Via Intensive Antennas in each conducting layer connected to optimized detection devices. Also includes reference device and isolation of damage.	PLSMA_A_N PLSMA_F_N PLSMA_A_P PLSMA_F_P PLSMA_V_N PLSMA_V_P	HCSTRESS VT CPV_3 CALC_VT HCI CPF_3 VPT IDVD IBIDIG MAXSTRS S PD_LINK

The next table lists some of the possible failure mechanisms detected using the PDQ-WLR test structures and software.

Plasma Damage Test Structures and Failure Mechanisms

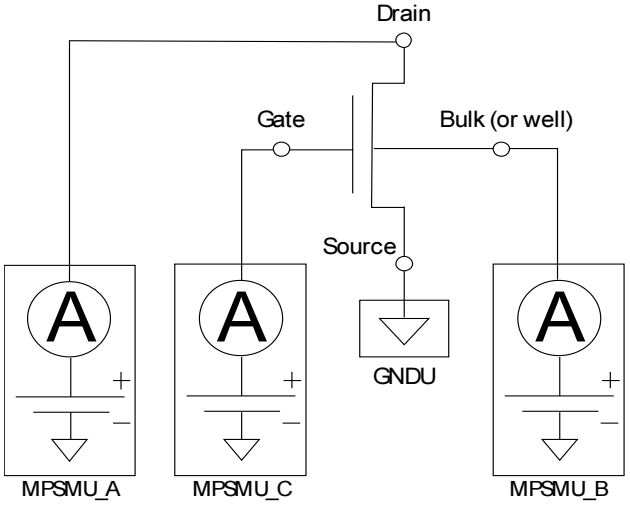
Test Structure Description	Failure Mechanisms	Tests
Plasma Damage	Polysilicon etch; M1, M2, M3, M4 etch; Photoresist ashing; S/D implant charging; Spacer Oxide Etch; Sputter Cleans; plasma CVD deposition	Hot Carrier & Charge Pumping

MAXSTRS_MOS

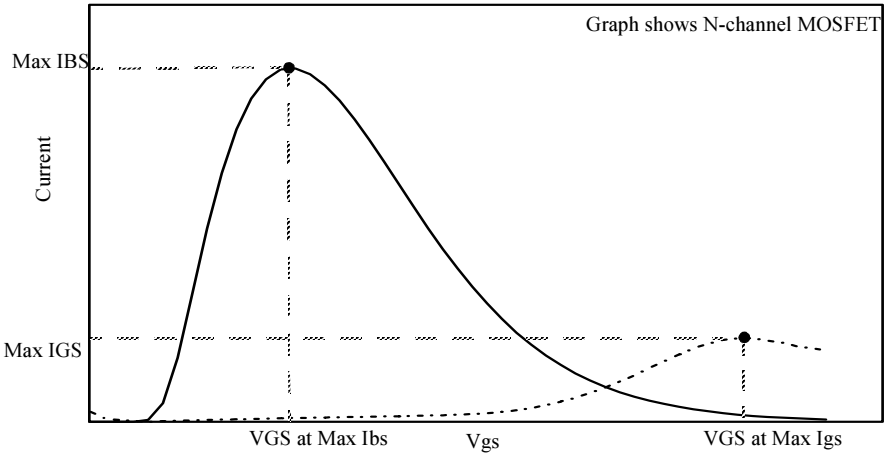
For a specified drain voltage (Vds), **MAXSTRS_MOS** estimates the gate voltage (Vgs) which will produce the maximum substrate current and maximum gate current.

This can be used to as a starting point to estimate worst-case gate voltage for a hot-carrier test.

Test Configuration



Electrical Characteristics



Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Vds†	Real	Volts	5	-100~100	Drain-to-Source bias
Vbs†	Real	Volts	0	-100~100	Bulk-to-Source bias
Vgs_min†	Real	Volts	0	-100~100	Start gate voltage
Vgs_max†	Real	Volts	6.5	-100~100	Stop gate voltage
Steps	Integer	——	101	2~1001	Number of Vgs sweep steps
Integ	Integer	——	2	1~3	SMU integration level 1: Short, 2: Medium, 3: Long
Plot$	String	——	"Y"††		Plot/Save data

† Values are for an N-channel MOSFET. Typically, Vds for a P-channel MOSFET would be less than zero, Vbs would be greater than zero and Vgs would be less than zero.

††A "Y" plots current versus voltage on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files.*

Device Variables

Var Name	Var Type	Comment/Description
Width	Real	um, Mask channel width
Length	Real	um, Mask channel length
Type$	String	(N/P), Channel type

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Maxib†	Real	Amps	Maximum bulk current
Vibmax	Real	Volts	Maximum Vgs at Maxib
Maxig†	Real	Amps	Maximum gate current
Vigmax	Real	Volts	Maximum Vgs at Maxig
Vds_max	Real	Volts	Drain voltage of measurement

† Returned as absolute value.

Algorithm Syntax

The syntax for calling the algorithm directly (that is, outside of Agilent SPECS) is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction to Hot Carriers* section.

Maxstrs_mos (Vds, Vbs, Vgs_min, Vgs_max, Steps, Integ, Plot$,
Pins(*), Width, Length, Type$,
Maxib, Vibmax, Maxig, Vigmax, Vds_max)

Theory of Operation

The *Introduction to Hot Carriers* section provides a general overview of hot-carrier-induced degradation.

For a specified drain voltage (Vds), **MAXSTRS_MOS** estimates the gate voltage (Vgs) which will produce the maximum substrate current and maximum gate current. For some technologies, stressing an N-channel MOSFET at a gate voltage near the maximum substrate current results in the maximum transistor degradation during hot-carrier stress. For other technologies, N-channel MOSFETs have their worst-case degradation at a gate voltage near the maximum gate current (maximum electron injection) or with a gate voltage around the threshold voltage (maximum hole injection). For P-channel MOSFETs, the bias conditions at maximum gate current can result in maximum transistor degradation during hot-carrier stress. The Vgs which produces the most damage should be determined for a given process. This is accomplished by hot carrier stressing a series of similar devices at the same drain voltage, and over a range of gate voltages from just above threshold to Vds.

The maximum Vds for stress is set by the physical limitations of the devices. Vds must be less than the reverse breakdown voltage of the drain-substrate diode, punchthrough voltage and snapback initiating voltage. A rule of thumb limits Ibs to below 20 microamperes per micron of device width. Another rule limits the ratio of Ibs to Ids to less than 15%. This ratio can be determined using the algorithm **IDIBIG_MOS**.

Testing Notes: At a high enough Vds, this measurement will damage the MOSFET due to channel hot carriers. It is recommended that a separate device should be used for this measurement and any subsequent hot-carrier experiment.

For low values of Vds, it may be necessary to set the Integ input variable to 3.

Data File Format: The input parameter Plot$ can be used to save the measured data in an ASCII file using this algorithm. The algorithm saves all the algorithm inputs and outputs into the file first. This is followed by the measured data saved in two sets. Each set of data contains two columns. The first column of each set contains the gate biases. The second column of the first data set contains the measured substrate currents at the corresponding gate biases while that of the second data set contains the measured gate currents. A row of column headers at the top of each data set identifies the parameters stored in individual columns. This format is shown in the following table:

Input parameters	
Output parameters	
VGS (V)	IB (A)
<data>	<data>
VGS (V)	IG (A)
<data>	<data>

Test Structure

This routine requires an N-channel or P-channel MOSFET. If possible, this device should have separate gate and/or substrate contacts. See also "Test Structures" in *Introduction to Hot Carriers* section.

Step-by-Step Action

1) Check for legal input values. If any are out of range, output variables are set to flag values 2.0E+30 and algorithm exits.
2) Connect SMUs as in Test Configuration diagram.
3) Force MOSFET terminal voltages.

4) Measure Ib curve

   a) Sweep gate voltage from Vgs_min to Vgs_max in the specified number of steps and measure Ib current at each step.

   b) Search Ib-Vgs curve to find maximum bulk current and corresponding voltage.

6) Measure Ig curve

   a) Perform second sweep gate voltage from Vgs_min to Vgs_max in the specified number of steps and measure Ig current at each step.

   b) Search Ib-Vgs curve to find maximum gate current and corresponding voltage.

7) Plot/save the data according to the value of Plot$.

8) Return Maxib, Vibmax, Maxig, Vigmax, and Vds_max.

Error Code Summary

This value is assigned to the output variable if an error occurred during algorithm execution:

Value	Meaning
2.0E+30	Input parameter error.
1.5E+30	Analysis not performed
+XE+32	Operating system error "X" occurred during execution of algorithm.

Compatibility with Previous Versions

This algorithm is compatible with previous HP BASIC test programs.

HCI_3_MOS

Conducts a user-specified hot carrier test. The user specifies the test using the **HCI_3_MOS Builder** to create the command file and time-stress table required as input to the routine. No programming is needed to develope sophisticated hot carrier tests.

There is no device curve to plot or test configuration needed.

Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
File_in	String	-	-	-	Full Path/Filename of command file

Device Variables

Var Name	Var Type	Comment/Description
None		

Output Variables

There are no output variables. The output data are stored in HC_LINK_2 and other algorithm areas.

Algorithm Syntax

The syntax for calling the algorithm directly (that is, outside of Agilent SPECS) is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction to Hot Carriers* section.

Hci_3_mos(File_in$,
Pins(*))

Theory of Operation

The *Introduction to Hot Carriers* section provides a general overview of hot carrier degradation.

HCI_3_MOS is designed to provide a convenient, generic interface for custom hot-carrier tests. No custom programming is required.

Methodology

Perform test(s) specified in the command file. Output results are linked to the database for further analysis. The **HCI_3_MOS Builder** is provided which makes it easy to input command files used by **HCI_3_MOS**. Thus, no programming is necessary to run complex hot-carrier tests. Several example command files are also provided for various technologies.

Data Storage in HCI_3_MOS: Provided the database is turned on (default), **HCI_3_MOS** uses **HC_LINK_2_MOS** to associate the stress data, the measurement data and the lifetime data. Data is stored in ASCII files which are spreadsheet compatible. The data consists of lists of input and output data, I-V curve data, and pointers. These pointers are used to relate the lifetime to the underlying data used to create it.

In **HCI_3_MOS**, there are 4 levels of data hierarchy. The lifetime data for each parameter is stored. Since the lifetime is created from calculated changes in device parameters, the percent change data is stored. Since this percent change is derived from raw device parameters, this data is also stored. And, since these raw device parameters are extracted from I-V curves, the I-V curves are stored if the Plot$ parameter is set for each algorithm.

For **HCI_3_MOS**, the lifetime data is stored in the /var/opt/WLR/database/data/measured/Hc_link_2_mos database area. The database contains a record for each lifetime specified in the command file. Each record consists of record number, date and time, test information (e.g. lot_name, wafer_name, die_name, module_name, device_name, wafer location, etc.) which is followed by the **HC_LINK_2_MOS** variables (Alg_name$, X_name$, X_value, Units$, EU, Change, Y_name$, Y_value, Vds, Vgs, Vbs, Ids, Ibs, Igs, File_ptr$, Width, Length, Type). Alg_name$ is the algorithm used to measure the lifetime (e.g. **VT_MOS**). X_name$ is the variable name assigned for this lifetime parameter in the command file (e.g. Idsat). X_value is the initial value of this parameter (e.g. pre-stress value of Idsat). Y_name$ is "Lifetime". The Y_value is the calculated lifetime value. Vds,Vgs and Vbs are the Stress conditions input from time-stress file, which is referenced by the command file. Ids, Ibs and Igs are the initial terminal currents at the stress voltage Vds, Vgs and Vbs.

File_ptr$ is the name of a pointer file that contains two entries. The first entry is the record number and filename containing the **LIFETIME_2_MOS** record for this particular **HC_LINK_2_MOS** record. The second entry contains a pointer to a file which in turn points to the parameter data recorded during the test. File_ptr$ always refers to a file in the /var/opt/WLR/database/data/hci/all_hci/lifetime_ptr area. The second entry in the File_ptr$ file always refers to a file in the /var/opt/WLR/database/data/hci/all_hci/stress_ptr area. The entries in this file refer to the PDQ-WLR database recordings, in the order in which they occurred. Each entry consists of a record number and data table file name, usually in the /var/opt/WLR/database/data/measured area. If an algorithm failed, or failed to make a database recording, the entry will consist of "0;*". Refer to the *Database Output* section for details regarding these files.

All database files are in comma-separated ASCII format, which can be queried for further study (the *Database Outputs* section shows how to transport these files into other tools.) For example, if the /var/opt/WLR/database/data/measured/Hc_link_2_mos data were transported to a spreadsheet, then the data can be sorted to locate all "Idsat" lifetime tests performed with the same stress conditions. If there is anomalous data, the pointer file information could be used to located the other device parameters or I-V curve.

Test structure

See "Test Structures" in the *Introduction to Hot Carriers* section.

Step-by-Step Action

1) Pretest the device using the algorithm sequence specified in the command file.

2) Iterate:

a) Call **STRSMEAS_2_MOS**, stress for the commanded time, measure Ids, Ibs, and Igs during the stress cycle.

b) Test the device using the algorithm sequence specified in the command file.

3) Stop test when.

a) specified maximum stress time is reached.

b) Stop criterion is met.

c) Device is bad (see below).

4)Call **LIFETIME_2_MOS** for all tracked parameters.

5)Perform final data linking using **HC_LINK2_MOS**.

Stop of HCI_3 Test Due to Device Failure

HCI_3 uses **STRSMEAS_2** and the measurement routines specified in the input file. If during the stress period, the terminal current (gate, substrate or drain) are higher than expected (or possible from the SMU), then the stress is removed and the test ends. The Bad_dut variable is also set to 1 in the **STRSMEAS_2** routine. The current limits are shown below in this table.

Drain	Bulk (or well)	Gate
10*1mA/um	10*100uA/um	Larger of 10*1uA/um or $10*10uA/um^2$

If the current through any of these terminals reaches the above value(or the SMU current capability) during the stress period, then the stress is stopped and the Bad_dut output variable is set to 1.

If a bad measurement condition (e.g. chuck or probe card become disconnected) was not detected during the stress period, **VT_MOS** (**S_MOS**) will detect this failure. The **HCI_3** test is also stopped under these conditions.

Error Code Summary

Refer to the MOSFET algorithms used during the hot carrier test.

Compatibility with Previous Versions

This algorithm will read previously generated input command files.

Example Tests

To ease the start of testing, we have provided example hot carrier test files. These tests may not work with every technology. If you are new to hot carrier testing, these files provide a starting point. All example files are accessible by the Load command in the **HCI_3_MOS Builder**.

Pretest

The pretests are designed to determine:

1. worst-case degradation mechanism (if this exists),
2. the type of damage at various Vgs stress conditions,
3. Vgs at peak damage to use in qualification (Qual) testing.

Unlike the current JEDEC tests, which specify a fixed Vgs, our pretest files vary Vgs during the stress to identify the worst case Vgs. Thus our pretests are faster and require fewer devices. They are designed to give a quick result and provide output which is easy to analyze in **PDQ-AT** (analysis tool). We do not guarantee that these techniques work with your technology, but they may provide a good starting point for technologies that have not been characterized.

These tests are based on the assumption that there is one dominant degradation mechanism. If these tests do not provide adequate results, then a series of constant voltage hot carrier stresses on separate devices with a fixed Vds and Vgs, varying across the devices is needed.

You may wish to specify a new database name for these tests to remove the need to filter the data later (making analysis with **PDQ-AT** easier).

In contrast to the JEDEC pretest methods, these tests, if successful with your technologies, will provide results at a fraction of the test time required for the JEDEC-compliant pretests. Typically, these pretests complete in a few minutes and require about 15 minutes of analysis time with **PDQ-AT**, compared with at least four hours of testing and an hour of analysis for the JEDEC methods.

The JEDEC tests require that a number of hot carrier tests be performed at discrete Vgs stress levels. The Vgs stress that creates the maximum damage is then used for qualification testing.

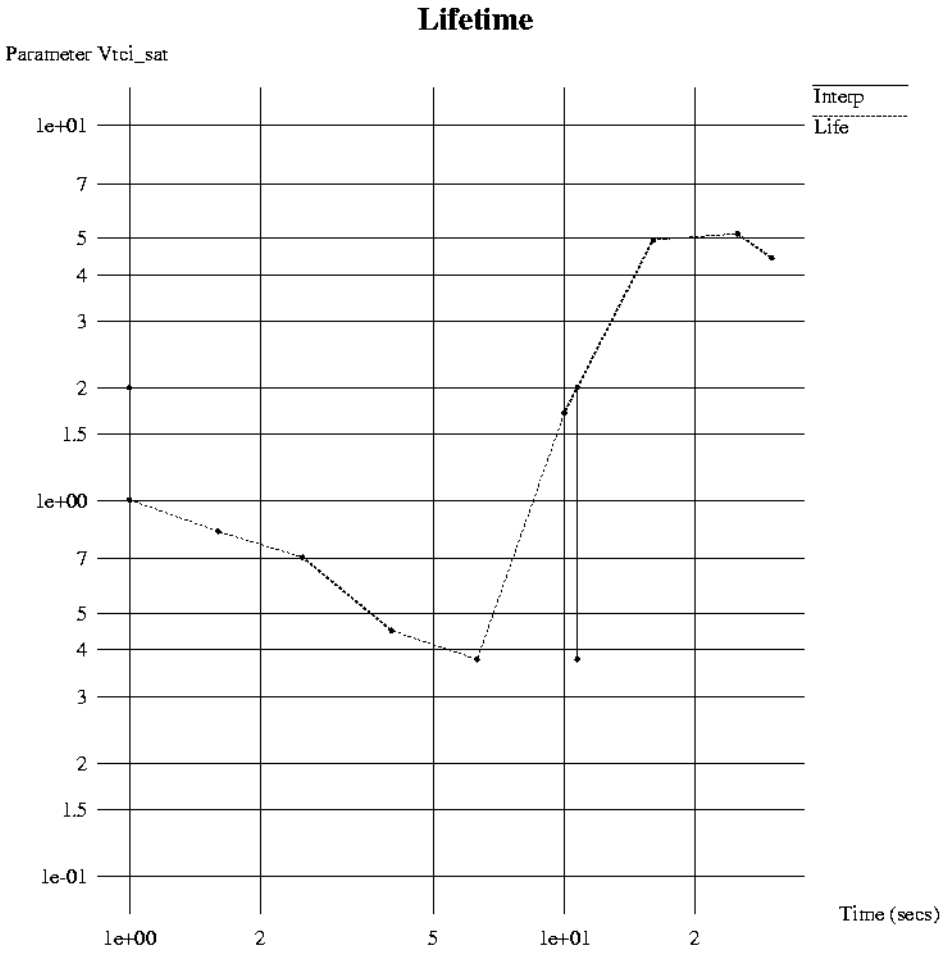
Finding Vgs at Peak Damage

In the following pretests, Vgs is swept, damaging the device incrementally. The resulting lifetime curves are not used for any lifetime predictions. Instead, these curves show the rate of change for damage to the device. The point along the curve with the highest absolute derivative represents the point at which maximum damage has occurred due to the Vgs stress voltage. Under ideal conditions, this curve will present the integral of the curve produced by point-plotting the results from the JEDEC-compliant pretest, and so the point of maximum damage peaks, and produces a sharp change in the derivative.

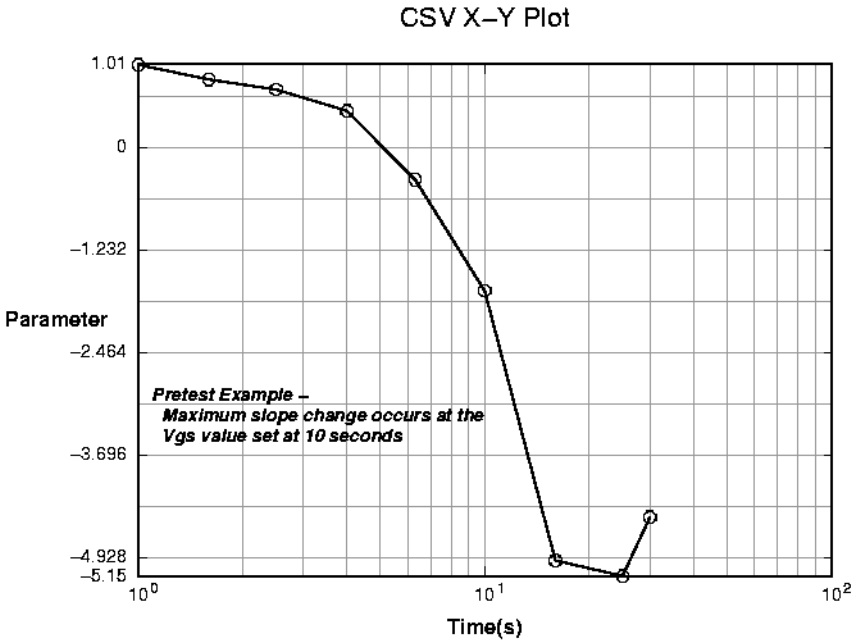
Once this point has been established, note the time at which it occurred on the lifetime plot. Using the **HCI_3_MOS Builder**, look up the Vgs value at this time; this is the value to use as the constant Vgs stress for your qualification tests.

The following images show an xgraph and csv plot example of the same HCI pretest data. Note that log-Y plotting takes the absolute values of the parameters to plot, and may therefore not show the the slope change as dramatically as the log-X, linear-Y case.

XGraph Plot Example



CSV Plot Example



If your device starts out with a high damage level, and shows a shallow curve on the lifetime plot, it should be swept from the opposite direction than the one provided in the sample file. An easy way to do this, using the **HCI_3_MOS Builder**, is to load the original command file, and then load in the time-stress file for the same technology of the opposite channel type. (There is always an "n" or a "p" in the file name to indicate the device type.) Re-save the file, and repeat the test.

Determining the Damage Mechanism

Use **PDQ-AT**, and retrieve the record from the database which contains the pretest results for Alg_name=**CPV_3_MOS**. Use the View tool on this dataset. Click on the linked file at the end of the pretest record (File_ptr) which contains the database keys for this test. The special **HC_LINK_2_MOS** point-plot viewer will be presented. Select the first **CPV_3_MOS** test record, and create a CSV plot. Next, overplot all successive **CPV_3_MOS** test data on the CSV plot. The result will show the

changes which occurred as the device was incrementally damaged. Refer to the PDQ-WLR training material for information on how to interpret these plots.

Notice

These pre-tests are not a substitute for careful characterization of hot carrier induced damage mechanisms on your technology, but in many cases can produce similar results to the JEDEC-compliant tests at a fraction of the time normally required.

pretest_n_6

This test file applies a constant Vds voltage with an increasing Vgs stress voltages from just below threshold to the Vds stress voltage. You may need to change the direction of Vgs stress for your technology or adjust the Vds stress value to achieve more or less damage during the sweep. The stress time is increases logarithmically during the Vgs sweep to account for the logarithmic time dependence of hot-carrier-induced damage.

VT_MOS linear parameters and **CPV_3_MOS** parameters are measured at the beginning and after each step in the Vgs sweep. Other parameters may also be included.

pretest_p_5

This example is similar to the n-channel test above, except the stress is started at high Vgs and is reduced to a Vgs close to threshold voltage. Again, the stress time proceeds logarithmically. **VT_MOS** in saturation and the **CPV_3_MOS** curve is measured after each step in the Vgs sweep. You may need to change the **VT_MOS** parameters to linear values for deep submicron technologies.

Qualification Tests

These are JEDEC standard 28 and 60 compliant constant voltage hot carrier tests which monitor a wide range of parameters. The voltage conditions (Vds and Vgs) may need to be changed to suit your technology. Proper selection of Vds and Vgs should be set based on the results of characterizing your technology using the pretests.

qual_ex_n_33

This example file is for n-channel MOSFET devices in a typical 0.4 um, 3.3 V technology. The drain stress voltage may need to be adjusted for your technology. The time-stress table shows that the stress proceeds logarithmically from 1 second up to maximum of 10,000 seconds. In the Save menu, you will see that we have chosen to stop the test after Id_lin has changed by 20%. You may need to increase or decrease this number, change the stop parameter value or type, or change the maximum stress time.

The Command List calls three different algorithms used during the measurement. **VT_MOS** is listed twice since two different measurement conditions (saturation and linear region) are performed. All the output parameters are saved and lifetime is calculated based on 10% change in parameters except for the **CPV_3_MOS** parameters. A power-law time-dependent model is assumed for all output values. For the **CPV_3_MOS** the lifetime is based on 1000% percent change in charge pumping current Ibmax and 200% change in V1o or V2o. For all measurements, we have chosen to save the Xgraph curve as well as the raw data. This allows the curves from which the parameters are extracted to be displayed. In particular, the command Save Algorithm Data allows the data to be plotted using the CSV tool in **PDQ-AT** or other tools such as a spreadsheet program. Setting these options can be very useful when analyzing the data but does increase the data storage.

qual_ex_n_25

This test is similar to the previous example except the stress voltages and the measurement algorithms have been set for a typical 0.25 um, 2.5 V technology. In this example, the test is stopped when the maximum stress time exceeds 10,000 seconds or Id_lin (linear drain current) exceeds 10%. This example assumes that the majority of damage is due to interface-state generation around or slightly below the peak substrate current.

qual_ex_n_50

This test is similar to the qual_ex_n_33 example except that the stress voltages and measurement algorithms have been adjusted for a typical 0.6 um, 5 V technology. In this example, the test is stopped when Gm_lin (linear maximum transconductance) exceeds 10%.

qual_ex_p_33

This example file is for p-channel MOSFET devices in a typical 0.4 um, 3.3 V technology. This example is based on electron trapping dominating the damage at peak gate current. The time-stress table shows that the stress proceeds logarithmically from 1 second up to maximum of 10,000 seconds. In the Save menu, the test is stopped after Id_sat has changed by 10%. You may need to increase or decrease this number, change the stop parameter value or type, or change the maximum stress time to suit your technology.

The Command List calls three different algorithms used during the measurement. **VT_MOS** is listed twice since two different measurement conditions (saturation and linear region) are performed. All the output parameters are saved and lifetime is calculated based on 10% change in parameters except for the **CPV_3_MOS** parameters. A logarithmic time-dependent model is assumed for all output values except Ibmax in **CPV_3_MOS**. For the **CPV_3_MOS** the lifetime is based on 1000% percent change in charge pumping current Ibmax and 200% change in V1o or V2o. For all measurements, we have chosen to save the Xgraph curve as well as the raw data. This allows the curves from which the parameters are extracted to be displayed. In particular, the command Save Algorithm Data allows the data to be plotted using the CSV tool in **PDQ-AT** or other tools such as a spreadsheet program. Setting these options can be very useful when analyzing the data but does increase the data storage.

qual_ex_p_25

This test is similar to qual_ex_p_33 except the stress voltages and the measurement algorithms are set for a typical 0.25 um, 2.5 V technology. This example is based on interface-state degradation dominating at peak hole-injection conditions and a power-law time dependence is assumed. The test is stopped when the maximum stress time exceeds 10,000 seconds or Gm_lin (maximum linear transconductance) exceeds 10%. Vgs and Vds stess conditions may need to be adjusted to suit.

qual_ex_p_50

This test is similar to qual_ex_p_33 except that the stress voltages and measurement algorithms have been adjusted for a typical 0.6 um, 5 V technology.

Preparing HCI_3_MOS for Agilent SPECS Use

Pretest and qualification test hot carrier command files need to be created prior to you first invocation of the Agilent PDQ-WLR Framework for Agilent SPECS.

1. Start the **HCI_3_MOS Builder**.
2. Load in the one of the pretest examples that matches your technology.
3. Make any changes desired, for example: fill in the correct width and length for your device.
4. When finished, save the file as *pretest*.
5. Load one of the qualification test examples.
6. Make changes.
7. Save as *qualification*.

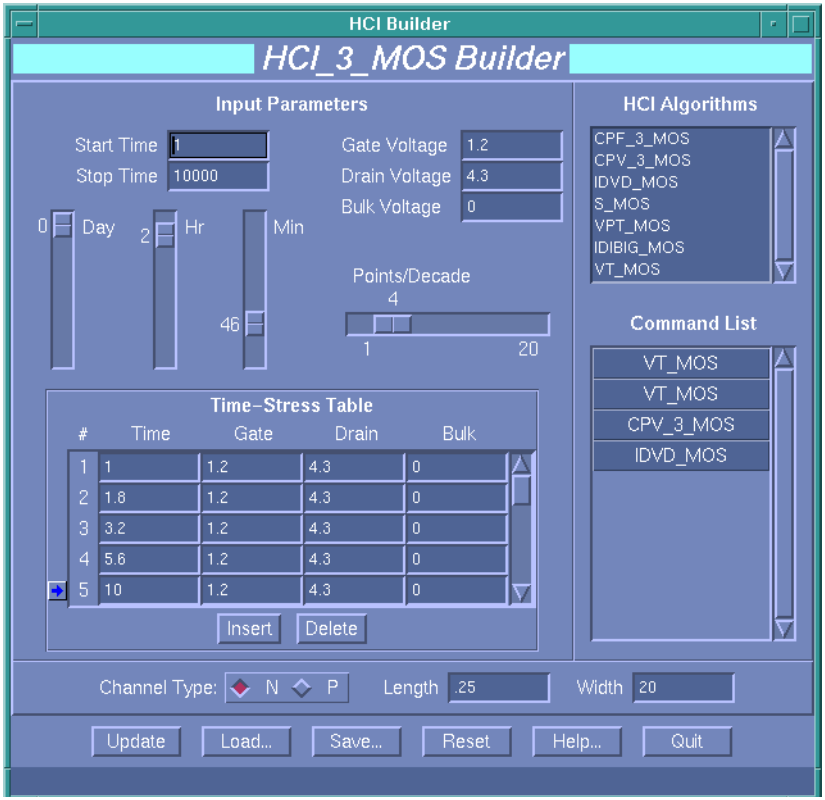
After running first pretest, the Agilent PDQ-WLR Framework, when commanded to, will perform a qual test presenting you with the qualification command file you just saved. At this point, update the time-stress table with the Vgs value selected from your pretest results.

HCI_3_MOS Builder

PDQ-WLR comes with the **HCI_3_MOS Builder** utility, which helps to create input command files for hot carrirer tests using **HCI_3_MOS**.

To launch the builder, type ***/opt/WLR/bin/xhci*** at the UNIX command prompt.

The HCI_3_MOS Builder



The Time Stress Table

You may build your Time-Stress table using this tool, or import your time-stress values from an external source as a CSV file.

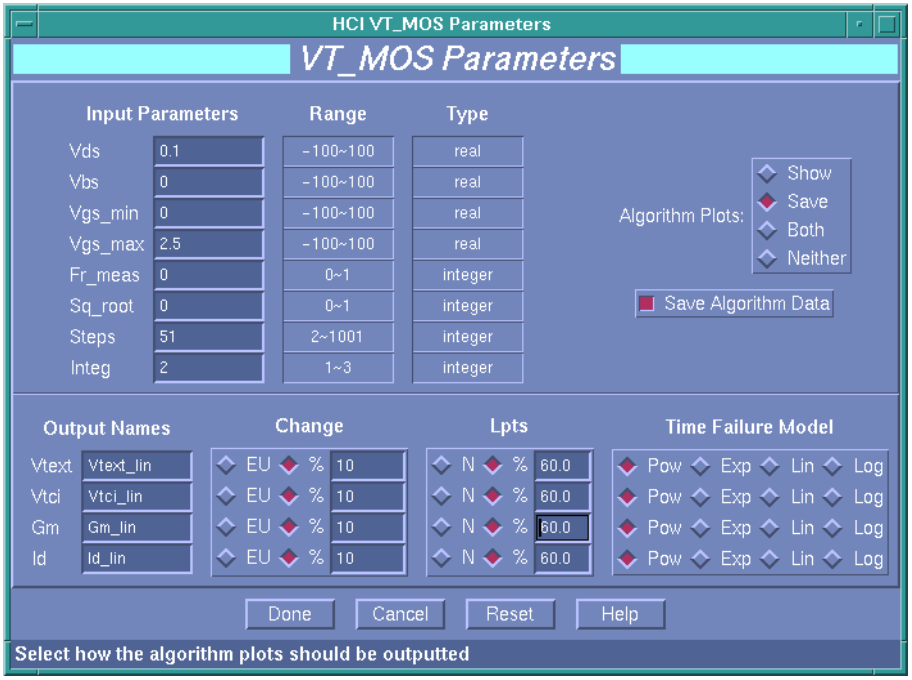
The Time Stress file is a 4 column, comma separated value (CSV) file displaying time, gate, drain, and bulk respectively. There are no header lines in the file and the file must be UNIX-formatted (without ^M's at the end of each line). You may use a spreadsheet program to create your table and save as a CSV file. The file must end with *.hcitime*.

How to Create an Input Command File for HCI_3_MOS

1. At the UNIX command prompt, type */opt/WLR/bin/xhci*. The **HCI_3_MOS Builder** window appears, see the above figure.
2. Type the Start Time value in the **Start Time** field. This field accepts a range from 0.1 to 1E6 (1,000,000).
3. Type the Stop Time value in the **Stop Time** field. This field accepts a range greater than 1 to 1E6 (1,000,000).

   Alternatively, you may use the **Day**, **Hr**, and **Min** slider bars to set the days, hours and minutes respectively. Click-drag the sliders to the desired Stop Time. As you do so, note how the Stop Time field value changes.
4. Type the Gate Voltage value in the **Gate Voltage** field.
5. Type the Drain Voltage value in the **Drain Voltage** field.
6. Type the Bulk Voltage value in the **Bulk Voltage** field.
7. Click-drag the **Points/Decade** slider to the number of tests you wish to perform per decade. 4 points per decade is usually sufficient.
8. Click Update to update the time-stress table.
9. Optionally, you may add or delete time-stress records individually.
10. Optionally, you may edit individual stress voltages.
11. Click the **HCI Algorithms** you desire to measure between stress intervals. The Algorithm window appears as below. The order of tests is as shown in the command list.

The HCI_3_MOS Builder Algorithm window



The above window shows an example Algorithm window for **VT_MOS**.

Each algorithm is slightly different in the number of Input and Output parameters. The software does not force you to enter an output variable and neither do you have to input all the variables — just the ones that are appropriate for your analysis.

12. Change any input parameters as needed. The parameters are pre-loaded with the PDQ-WLR defaults. Defaults are all listed in this manual in the appropriate algorithm.

13. Select if you desire to have Xgraph (executable graphs) of the algorithm to be saved, displayed, both or neither during the test.

14. If you desire to have CSV files created from the test, click the **Save Algorithm Data** button (recommended).

15. Type the desired output variable name in the in appropriate **Output Name** field. If no output name is entered, lifetime is not calculated for that variable.

16. Select EU (Engineering Units, e.g. V, A) or % as lifetime criteria.

17. Enter the **Change** % desired (e.g. 10%) or **E.U.** (e.g. 0.1 for 100mV in Vtext).

18. For Lifetime fits requiring extrapolation to the **Change** criterion, you may choose how many points (Lpts) of the record to use for a least-squares fit. The default is **60%** of the measured points. You may override this. Select either % or N (where N is the actual number of points to use regardless of the number of measured points). When N is selected, the lifetime calculation is based on the minimum of your specified **N** or the total number of actual measured points.

19. Enter **Lpts** desired (e.g., 10 for 10% or 10 for 10 points).

20. Select the **Time Failure Model** desired.

    Pow models on a Log X, Log Y scale. Recommended for nMOS and advanced pMOS.
    Exp models on a Linear X, Log Y scale
    Lin models on a Linear X, Linear Y scale
    Log models on a Log X, Linear Y scale. Recommended for >0.25 um pMOS.

21. Click **Done**. The algorithm window closes, and is added to the command list.

Repeat steps 8 to 21 as necessary.

The chosen measurement algorithm appears in the **Command List**. The order of measurement is as shown in the Command List.

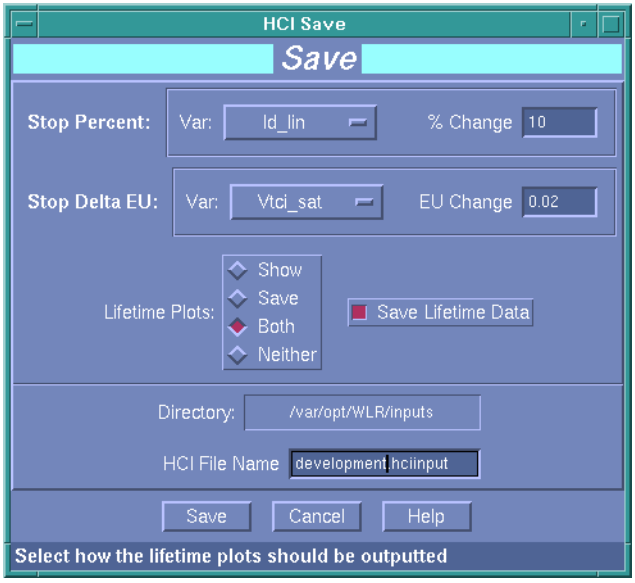
You may delete algorithms from the Command List by centering the mouse icon over the undesired algorithm button and pressing the ***D*** key on the keyboard.

To move an algorithm, click-drag the algorithm with the middle mouse button to the desired location in the Command List.

22. When all algorithms are loaded in the Command List, click the **Update** button.

23. Select the **Type**. You choice is N-channel or P-channel.

The HCI_3_MOS Builder Save window



24. Type the **Width** and **Length** in their respective test fields.
25. Click the **Save...** button. The save widow opens.
26. If you desire to set a Stop Variable, stopping the test whenever a certain value is passed, click-drag the **Stop Percent** and enter the **% Change** necessary to stop the test. You can also specify a **Stop Delta EU.** If both a **Stop Percent** and a **Stop Delta EU** are specified then the test is stopped when the first one is reached. Only variables which are tracked for lifetime calculation can be used as the Stop Variable.
27. Select if you desire to have Xgraph (executable graphs) of the **LIFETIME_2_MOS** to be saved, displayed, both or neither during the test.
28. If you desire to have CSV files created from the test, click the **Save Lifetime Data** button (recommended).
29. Enter the desired name of this command file in the **HCI File Name** field.

    Names can not have blank spaces, but they can hold underscores (_).
30. Click the **Save** button.

Your new Input Command List and Time Stress Table is saved to the */var/opt/WLR/inputs* directory. The HCI Builder creates two files when it saves: Time Stress Tables (***filename***.*hcitime*) and the Input Command List (***filename***.*hciinput*).

The **HCI_3_MOS Builder** does not add points between .1 and 1 second. You may add time-stress records between .1 and 1 manually with the Insert button.

It may be physically difficult to accurately stress and measure up to 20 logarithmically spaced points in the decade from .1 to 1 second for your situation, so the tool is designed to let you specify this input range yourself.

Please note that the drain, substrate and gate current is measured during each stress interval. So, depending on the current level being measured, the total stress time may exceed specified time. This is usually a concern for stress intervals much less than one second. In all cases, however, the actual stress time is recorded for accurate time degradation modeling.

How to Load a Input Command List or a Time Stress Table

1 Click the **Load...** button. The builder window appears.

The HCI_3_MOS Builder Load window

2 Use the **Data Path** popup, the **Up-Directory** button and the directory browser to locate the file you wish to import. By default the directory browser opens to */var/opt/WLR/inputs*.

If you are loading a *.hciinput* file, you must have the accompanying *.hcitime* file.

If you desire to load a different Time Stress Table to an Input Command File, load the Input Command File (*.hciinput*) first, then load the Time Stress Table (*.hcitime*). Use the filter popup to differentiate between *.hcitime* and *.hciinput* files.

If you cannot find your file, ensure your file is named appropriately.

3 Click on the file you desire to open and click the **Load** button.

Alternatively, you may simply double-click the desired file.

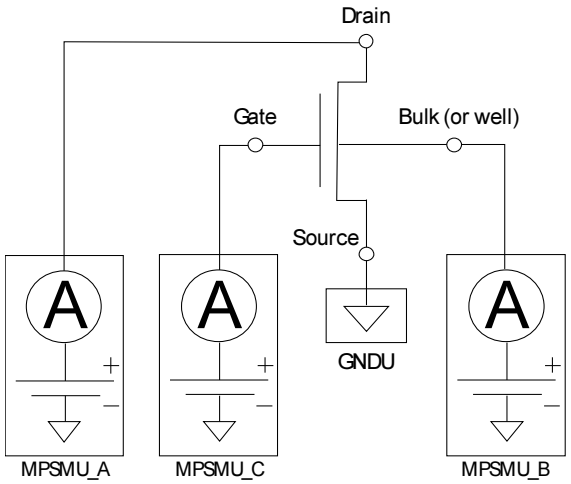
The file is loaded into the **HCI_3_MOS Builder**.

STRSMEAS_2_MOS

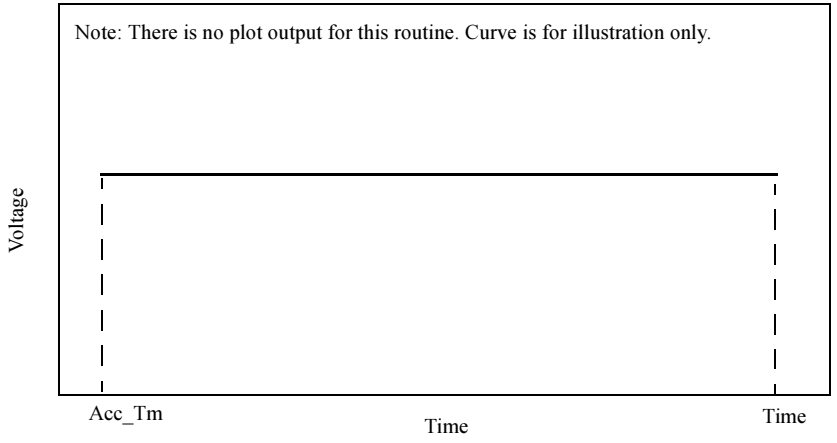
Apply gate, drain and bulk voltage to transistor for hot-carrier-induced degradation stress in compliance with JEDEC standards JESD28 and JESD60 for hot carrier tests.

STRSMEAS_2_MOS algorithm applies user supplied stress voltages to a **MOSFET** for a specified period of time to induce hot-carrier damage to the device. It also measures the drain (Ids), substrate (Ibs), and gate (Igs) currents during this time period.

Test Configuration



Electrical Characteristics



Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Time	Real	Secs	1.0	0.1~1E6	Time of stress
Vgs†	Real	Volts	2	-100~100	Gate-to-Source stress voltage
Vds†	Real	Volts	5.5	-100~100	Drain-to-Source stress voltage
Vbs†	Real	Volts	0	-100~100	Bulk-to-Source stress voltage
Acc_tm	Real	Secs	0	0~1.0E6	Previous accumulated stress time

† Values are for an N-channel MOSFET. Typically, Vds for a P-channel MOSFET would be less than zero,
Vbs would be greater than zero and Vgs would be less than zero.

Device Variables

Var Name	Var Type	Comment/Description
Width	Real	um, Mask channel width
Length	Real	um, Mask channel length
Type$	String	(N/P), Channel type

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Tstress	Real	Secs	Accumulated Time
Ids	Real	Amps	Drain Current
Ibs	Real	Amps	Substrate Current
Igs	Real	Amps	Gate Current
Bad_dut	Integer		Bad DUT flag, 1=bad, 0=OK

Algorithm Syntax

The syntax for calling the algorithm directly (that is, outside of Agilent SPECS) is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction to Hot Carriers* section.

STRSMEAS_2_MOS(Time, Vgs, Vds, Vbs, Acc_tm,
Pins(*), Width, Length, Type$,
Tstress, Ids, Ibs, Igs, Bad_dut)

Theory of Operation

The *Introduction to Hot Carriers* section provides a general overview to hot-carrier induced degradation in MOSFETs.

Algorithm **STRSMEAS_2_MOS** applies a constant user-specified voltage stress and measure the gate, drain, and bulk current of a MOSFET. The voltage is applied for a user-specified time or the time required to measure the currents, whichever is greater. The initial current values are sometimes used to scale the hot-carrier induced lifetime at operating conditions from accelerated stresses. The ratio of Ibs/Ids during stress also indicates the change in carrier heating.

This algorithm supplies the stress component of the stress/measure cycle described in the *Introduction to Hot Carriers* section. After the stress, transistor parameters are measured and compared to the initial unstressed values. These parameters can include but are not limited to the linear drain current (Ids), maximum transconductance (Gm), threshold voltage (Vtci), punchthrough voltage (Vpt), subthreshold slope (S) as well as charge pumping current (Icp).

The amount of hot-carrier-induced degradation is exponentially dependent on the drain voltage and increasing the drain voltage during stress will reduce test time. However, the maximum drain voltage is determined by the technology and test structure design. A "rule of thumb" limits the initial substrate current (Ibs) to less than 20 microamperes per micron of device width which can be determined using the **MAXSTRS_MOS** algorithm. Another rule limits the ratio of initial substrate current to drain current to less than 15% which can be determined with the **IDIBIG_MOS** algorithm.

The hot-carrier-induced degradation also strongly correlates to the gate voltage during stress. An initial value of Vgs can be determined using the **MAXSTRS_MOS** algorithm or the **HCI_3_MOS** pretests.

For N-channel MOSFETs in a digital process, parameters typically monitored during stress are Ids, Gm, Vtci, S, and Icp. Stress will cause the |Vtci |, |Icp |, and |S| to increase while |Ids| and |Gm| will decrease. For lightly doped drain (LDD) devices, the hot-carrier-induced threshold voltage shift is usually negligible, and the Gm or Ids change dominates the degradation. For P-channel MOSFET devices, |Vpt | decreases. For devices above 0.5 µm channel length, |Ids| and |Gm| increase while for devices around 0.25 µm, they decrease, and |Vtci | may increase or decrease.

Test Structure

See "Test Structures" in the *Introduction to Hot Carriers* section.

Step-by-Step Action

1) Check for legal input values. If any are out of range, output variables are set to flag values 2.0E+30 and algorithm exits.
2) Connect SMUs as in Test Configuration diagram in order specified by JESD28 and JESD60.
3) Calculate Stress_time = Time - Acc_tm
4) Force stress voltages.
5) Start timer.
6) Measure Ids, Ibs, and Igs.
7) If current exceeds compliance or the current limits then stop test and set Bad_dut=1
8) Wait until elapsed time equals or exceeds Stress_time.
9 Return accumulated stress time, absolute value of Ids, Ibs, Igs, and Bad_dut.

Bad Device Determination

During the stress, the current is limited to the SMALLER of the SMU capability or the following:

Drain	Bulk (or well)	Gate
10*1mA/um	10*100uA/um	Larger of 10*1uA/um or10*10uA/um^2

Error Code Summary

This value is assigned to the output variable if an error occurred during algorithm execution:

Value	Meaning
2.0E+30	Input parameter error.
1.5E+30	Analysis not performed
+XE+32	Operating system error "X" occurred during execution of algorithm.

Compatibility with Previous Versions

STRSMEAS_MOS is provided to act as a parametric translation shell to maintain compatability with HP BASIC test programs written for the obsolete **STRSMEAS_MOS**. This algorithm, which calls **STRSMEAS_2_MOS**, is briefly described below.

STRSMEAS_MOS

Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Time	Real	Secs	1.0	0.1~1E6	Time of stress
Vgs†	Real	Volts	2	-100~100	Gate-to-Source stress voltage
Vds†	Real	Volts	5.5	-100~100	Drain-to-Source stress voltage
Vbs†	Real	Volts	0	-100~100	Bulk-to-Source stress voltage
Acc_tm	Real	Secs	0	0~1.0E6	Previous accumulated stress time

† Values are for an N-channel MOSFET. Typically, Vds for a P-channel MOSFET would be less than zero,
Vbs would be greater than zero and Vgs would be less than zero.

Device Variables

Var Name	Var Type	Comment/Description
Width	Real	um, Mask channel width
Length	Real	um, Mask channel length
Type$	String	(N/P), Channel type

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Tstress	Real	Secs	Accumulated Time
Ids	Real	Amps	Drain Current
Ibs	Real	Amps	Substrate Current

Var Name	Var Type	Var Unit	Comment/Description
Igs	Real	Amps	Gate Current

Algorithm Syntax

The syntax for calling the algorithm directly (that is, outside of Agilent SPECS) is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction to Hot Carriers* section.

STRSMEAS_MOS (Time, Vgs, Vds, Vbs, Acc_tm,
Pins(*), Width, Length, Type$,
Tstress, Ids, Ibs, Igs)

Methodology

This algorithm is a translation shell which calls the **STRSMEAS_2_MOS** algorithm to cinduct the test. This shell maintains compatibility with existing HP BASIC test programs by changing the following input/output parameters in the call from **STRSMEAS_MOS** to **STRSMEAS_2_MOS**.

Input Parameters

No change

Output Parameters

Bad_dut: New parameter, not returned to **STRSMEAS_MOS**

HCI_MOS

This routine is provided for compatibility with previous versions of PDQ-WLR and is no longer a supported component. Custom programming efforts to support hot carrier testing are no longer necessary with PDQ-WLR; users are encouraged instead to use the new HCI_3_MOS and the HCI_3_MOS Builder tools.

During installation, PDQ-WLR took the following actions:

1. Installed /opt/WLR/lib/HCI_MOS_Lib. This file includes the original HCI_MOS and the required support components formerly located in /opt/WLR/lib/WLRLibSupport.
2. Made loading HCI_MOS_Lib the default action of WLR.config.
3. Detected if you had a previous copy of /opt/WLR/BASIC/HCI_MOS, and if so, moved it to /opt/WLR/local/HCI_MOS.Obsolete. (Use this file only to recover edits to HCI_MOS which you may have made at your site; if you've never modified HCI_MOS before, you may safely delete this file.)
4. Installed a special placeholder file in /opt/WLR/BASIC/HCI_MOS; this is only used to satisfy any existing scripts which you may have built to call makewlrprog (HP BASIC users only) that reference HCI_MOS.
5. Installed the /opt/WLR/local/ALL_HCI and all_hci.input files as described in this section.

If you no longer need HCI_MOS, you can easily retire it from you system to save execution space. To do this, change the WLR.config file so that instead of loading HCI_MOS_Lib, you instead load /opt/WLR/lib/No_HCI_MOS_Lib. Doing so ensures that all program references in the PDQ-WLR library are satisfied.

Provides interface between Agilent SPECS and user-written hot carrier algorithms. It can also be used in a stand-alone HP BASIC environment.

Two example algorithms, ALL_HCI and VT1_HCI, are provided for use with **HCI_MOS**. The ALL_HCI algorithm performs a detailed hot-carrier test. The user can also quickly customize the hot carrier test using ALL_HCI with limited programming. Data can also be stored in a relational format, which speeds data analysis.

There is no device curve to plot or test configuration needed.

Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Vgs†	Real	Volts	2	-100~100	Voltage stress on gate
Vds†	Real	Volts	5.5	-100~100	Voltage stress on drain
Vbs†	Real	Volts	0	-100~100	Voltage stress on bulk
Progfile$	String	——	——	" "	Filename of HP BASIC file that contains one or more custom algorithms
Alg$	String	——	——	" "	Custom algorithm subprogram to use, located in Progfile$

† Values are for an N-channel MOSFET. Typically, Vds for a P-channel MOSFET would be less than zero,
Vbs would be greater than zero and Vgs would be less than zero.

Device Variables

Var Name	Var Type	Comment/Description
Width	Real	um, Mask channel width
Length	Real	um, Mask channel length
Type$	String	(N/P), Channel type

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Hci_name$	String	——	Names of parameters
Hci_data	Real Array (1:3000)	——	Measurement result
Hci_strs	Real Array (1:200)	——	Stress conditions
Hci_num	Real	——	Number of entries in Hci_data
Hci_stat	Real	——	Return status 0 = OK

Algorithm Syntax

The syntax for calling the algorithm directly (that is, outside of Agilent SPECS) is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction to Hot Carriers* section.

Hci_mos (Vgs, Vds, Vbs, Progfile$, Alg$,
Pins(*), Width, Length, Type$,
Hci_name$, Hci_data, Hci_strs, Hci_num, Hci_stat)

Theory of Operation

The *Introduction to Hot Carriers* section provides a general overview of hot carrier degradation.

HCI_MOS is designed to provide a convenient, generic interface for a custom hot-carrier tests. If either the ALL_HCI or VT1_HCI example algorithm is used to perform hot carrier tests under **HCI_MOS**, the user can skip the **Methodology** section below and proceed directly to the **Example Algorithms**.

Methodology

The name of the custom HCI test is passed to this algorithm which loads it into memory and executes it. Because of the generic nature, each custom routine must have the same calling parameters as described below:

SUB Hcicustom_mos (Vgs, Vds, Vbs, Pins(*), Width, Length, Type$, Hci_name$, Hci_data(*), Hci_strs(*), Hci_num, Hci_stat)

Variable	Type	In/Out	Comment/Description
Vgs	Real	In	Voltage stress on gate (V)
Vds	Real	In	Voltage stress on drain (V)
Vbs	Real	In	Voltage stress on bulk (V)
Pins	Integer Array	In	Pins
Width	Real	In	Device parameter
Length	Real	In	Device parameter
Type$	String	In	Device parameter
Hci_name$	String	Out	List of parameter names
Hci_data	Real Array	Out	Table of parameter results
Hci_strs	Real Array	Out	Table of stress conditions
Hci_num	Real	Out	Entries in Hci_data
Hci_stat	Real	Out	Status of hci_getparm

Procedures: Three procedures are used to handle the data. All arrays are 1 based.

1) To save measurement data
Hci_savedata (Hci_name$, Hci_data(*), Hci_num, Stress_no, "name", Result)

2) To save stress conditions
Hci_savestrs (Hci_strs(*), Stress_no, Time, Vgs, Vds, Vbs)

3) To extract data for the **LIFETIME_MOS** algorithm
Hci_getparm (Hci_name$, Hci_data(*), Hci_num, Stress_no, "name", Parm_array(*), Time_array(*), Narray, Stat)

Variable	Type	Comment/Description
Hci_name$	String	List of parameter names
Hci_data	Real Array	Table of parameter results
Hci_strs	Real Array	Table of stress conditions
Hci_num	Real	Entries in Hci_data
Stress_no	Real	Stress number (0 … n)
"name"	String	Name of parameter
Result	Real	Value of Parameter
Time	Real	Time of stress (secs)
Vgs	Real	Voltage stress to gate (V)
Vds	Real	Voltage stress to drain (V)
Vbs	Real	Voltage stress to bulk (V)
Parm_array	Real Array	Vector of results for a parameter
Time_array	Real Array	Vector of stress times for a parameter
Narray	Real	Size of vector (base 1)
Stat	Real	Status of hci_getparm

Data Tables: One advantage of the HCI Method over the Sequence Method is handling of the output results. In the sequential method, the same algorithm is listed for each stress and the output variables usually need to be

uniquely named. This becomes a chore and error prone if there are several stresses. The **HCI_MOS** algorithm handles the data housekeeping in several data tables. These data tables are listed below.

Data Tables

Hci_name$

Name of Parm
ALG NAME
Name 1
Name 2
...

This list contains the name of the custom algorithm and names of the parameters which were saved by the custom algorithm. Each entry in the list is 16 characters long. Entries are located using character positions.
The maximum length of this string is 255 characters (or 14 parameter names).

Hci_data Table

Stress No	Name Index	Result Value
1	17	2.54
1	33	2.22
2	17	2.66
2	33	2.31

This structure contains the measured parameter values.

Hci_strs Table

Time	Vgs	Vds	Vbs
0	—	—	—
1.0	.3	.6	.9
5.0	.4	.65	.92

This structure contains the stress conditions used. Note, entry #1 is stress ID of 0 (pre-stress measurement).

If the algFlags variable "Save HCI Tables" is on (which is the default), then the data table information is saved in a spreadsheet-compatible ASCII format. This file is located in /var/opt/WLR/database/data/hci/stress_tables/ directory. The filename will begin with "Hci_out_" and end with a ".csv". The

remainder of the filename is the date and time of the test. In this file the Hci_name$ table and Hci_data table are combined into the one table. The second table is the Hci_strs table.

Example Algorithms

There are two example hot carrier algorithms for use with **HCI_MOS**: ALL_HCI and VT1_HCI. Both of these algorithms can perform a JESD28 and JESD60 compliant hot carrier test. In additional to the information below, a README file is located in /opt/WLR/newconfig/samples/ . The algorithm source files are also located in this area.

ALL_HCI Algorithm: This algorithm performs a sophisticated hot-carrier test and determines the lifetime of user-specified device parameters.

A wide range a hot-carrier tests can be performed using ALL_HCI. Changes to the stress times and measurement algorithms can be made without programming. An ASCII input file is read by ALL_HCI and sets the stress time and measurement algorithms (e.g. **VT_MOS**, **CPV_MOS**, etc.) during the test.

The supplied input file "all_hci.input" is shown below. This file illustrates a hot-carrier characterization in development or qualification. With this file, ALL_HCI performs a N-channel hot-carrier test and calculates the lifetimes for 10 device parameters. As shown in the listing below, a linear **VT_MOS** is performed followed by a saturation **VT_MOS** measurement. Charge pumping is performed (with **CPV_3_MOS**) and an **IDVD_MOS** curve is measured. The drain current, bulk current and gate current is then measured at the stress voltages using **IDIBIG_MOS**. The device is then stressed with **HCSTRESS_MOS** for the user-specified cumulative time. The same measurements are then performed. This stress/measurement cycle is repeated until the maximum time is reached.

ALL_HCI calculates the percent change in parameters at each stress time. This data is used by **LIFETIME_MOS** to calculate the lifetime. In ALL_HCI, the Lpts variable is set to 60% of the stress-data. This algorithm also assumes a power-law for a N-channel device and logarithmic for a P-channel device. The user can change these parameters by modifying ALL_HCI.

Supplied all_hci.input file:

```
#
# all_hci.input
```

```
#
# (c) Copyright 1998 Agilent Technologies  All rights reserved.
# (c) Copyright 1998 Sandia Technologies, Inc.  All rights reserved.
#
# WLR Source Version: $Revision: 1.4 $
# Last revised date:  $Date: 2000/01/26 20:49:29 $
# Source module name: $RCSfile: all_hci.input,v $
#

# Provides inputs to ALL_HCI.
# Blank lines, lines beginning with "#", and all spaces are
# ignored.
# Must be located in /var/opt/WLR/inputs/all_hci.input for
# the distributed ALL_HCI to operate correctly.


#
# Specify the Lifetime Change percentage to be calculated.
# The same Change is applied to all device parameter
# Lifetime calculations.
#

Change=10

#
# Specify how you want to handle Lifetime Plot$ data.
# In this example, graphing is set to generate
# graph files in batch mode (not plotted to
# screen in real-time) and a Csv file is also
# saved.  In each case, the default PDQ_WLR
# data areas (graphs and csv_files located
# under /var/opt/WLR/database/data) are used,
# along with default file names.  Filenames
# will be "algorithm name"_"date/time" followed
# by ".xgr" for graph files, and ".csv" for the
# spreadsheet-readable data files.

Plot$="S;Y"

#
# The above only applies to Lifetime plotting.
# Other algorithms have separate Plot$ entries,
# below.  Not coincidentally, the test algorithms
# are using the same Plot$ specification as
# Lifetime, allowing PDQ-WLR to handle all data
# management.  If you prefer other storage locations
```

```
# for this data, there is no danger of changing
# these entries.  PDQ-WLR is configured to follow
# data anywhere on your system.
#

# Provide a list of stress times for Hcstress_mos.
# A maximum of fifty (50) are accepted, but they must be
# all on the same line.  Your editor may produce strange
# looking results for long lines, but the code will
# operate correctly.  If using the HP-UX Vue editor, be
# sure to turn word-wrapping OFF.
#

Time=0, 0.7, 1, 1.6, 2.8, 5.2, 10, 19.6, 38.8, 77.2, 154

#
# A command line is required for each MOS algorithm to run
# during the test.  The following hot-carrier algorithms
# are supported:
#
# VT_MOS
# S_MOS
# DOP_MOS
# IDVD_MOS
# CPV_3_MOS
# CPF_3_MOS
# IDIBIG_MOS
#
# In this section, specify the algorithms you want to run, in the
# order you want to run them.  For each algorithm, specify the
# values to use for the input parameters.  Device and terminal
# parameters will passed through via HCI_MOS.  Notice that the
# Plot$ inputs are quoted.
#
# Output variables are not specified in the algorithm command
# line.  Instead, ALL_HCI handles it internally using the
# "Lifetime=" entries.
#
# To include a variable in the lifetime analysis, enter a
# "Lifetime=" line just below the algorithm call specification.
# The Lifetime parameters are associated with the ORDER of
# algorithm outputs, not their names.  In this way, you can
# use the same algorithm more than once, but store separate
# outputs for Lifetime analysis.  You do not have to specify
# Lifetime= line for all algorithms; notice that in this example,
# it was not used for Idibig_mos.  You do not have to use all
```

```
# of an algorithms outputs, but if you use the Lifetime= line,
# then you must provide placeholders.  Suppose that we were
# only interested in the first and third of the four outputs
# from Vt_mos.  In that case, the Lifetime= line would look like:
# Lifetime="Vtext_sat", "", "Gm_sat", ""
#
# A system constraint limits you to a total of fourteen (14)
# Lifetime parameters.  If you specify more than that, ALL_HCI
# will quietly ignore any beyond the first 14.
#
# No more than 32 Command or Lifetime lines accepted.

#
# Example of JEDEC hot-carrier test for N-type device
#

Vt_mos(3.3, 0, 0, 3.3, 0, 1, 101, 2, "S;Y")
Lifetime="Vtext_sat", "Vtci_sat", "Gm_sat", "Id_sat"

Vt_mos(0.1, 0, 0, 3.3, 0, 0, 101, 2, "S;Y")
Lifetime="Vtext_lin", Vtci_lin", "Gm_lin", "Id_lin"

Cpv_3_mos(0, 0, 2, 1.E6, 100, 100, -4, 1, 25, 2, "S;Y")
Lifetime="Ibmax","",""

Idvd_mos(0, 3.3, 50, 1, 3.3, 2, 0, 10, 2, "S;Y")
Lifetime="Gd"

#
# Example of JEDEC hot-carrier test for P-type device
#

# Vt_mos(-3.3, 0, 0, -3.3, 0, 1, 101, 2, "S;Y")
# Lifetime="Vtext_sat", "Vtci_sat", "Gm_sat", "Id_sat"

# Vt_mos(-0.1, 0, 0, -3.3, 0, 0, 101, 2, "S;Y")
# Lifetime="Vtext_lin", Vtci_lin", "Gm_lin", "Id_lin"

# Cpv_3_mos(0, 0, 2, 1.E6, 100, 100, -2, 3, 25, 3, "S;Y")
# Lifetime="Ibmax","",""

# Idvd_mos(0, -3.3, 50, -1, -3.3, 2, 0, 10, 2, "S;Y")
# Lifetime="Gd"

#
# Always perform Idibig_mos last,  because it produces an additional
```

```
# stress which would contaminate the results of the above measurements.
#
# Note that the Vds, Vgs, and Vbs inputs are required inputs for the
# ALL_HCI pre-processor,  but are not used.  Any entry for these
# variables will be overridden by the HCI_MOS entries.  For conve-
nience,
# leave these at 0.
#

Idibig_mos(0., 0., 0., 0, 2)

# End of file (this line not required)
```

Data Storage in ALL_HCI: Provided the database is turned on (default), ALL_HCI uses **HC_LINK_MOS** to associate the stress data, the measurement data and the lifetime data. Data is stored in ASCII files which are spreadsheet compatible. The data consists of lists of input and output data, I-V curve data, and pointers. These pointers are used to relate the lifetime to the underlying data used to create it.

In All_HCI, there are 4 levels of data hierarchy. The lifetime data for each parameter is stored. Since the lifetime is created from calculated changes in device parameters, the percent change data is stored. Since this percent change is derived from raw device parameters, this data is also stored. And, since these raw device parameters are extracted from I-V curves, the I-V curves are stored if the Plot$ parameter is set for each algorithm.

For ALL_HCI, the lifetime data is stored in the /var/opt/WLR/database/data/measured/Hc_link_mos/ database area. The database contains a record for each lifetime specified in the "all_hci.input" file. Each record consists of record number, date and time, test information (e.g. lot_name, wafer_name, die_name, module_name, device_name, wafer location, etc.) which is followed by the **HC_LINK_MOS** variables (Alg_name$, X_name$, X_value, Y_name$, Y_value, Vds, Vgs, Vbs, Ids, Ibs, Igs, Width, Length, Type, File_ptr$). Alg_name$ is the algorithm used to measure the lifetime (e.g. **VT_MOS**). X_name$ is the variable name assigned for this lifetime parameter in the all_hci.input file (e.g. Idsat). X_value is the initial value of this parameter (e.g. pre-stress value of Idsat). Y_name$ is "Lifetime". The Y_value is the calculated lifetime value. Vds,Vgs and Vbs are the stress conditions input from **HCI_MOS**. Ids, Ibs and Igs are the initial terminal currents at the stress voltage Vds, Vgs and Vbs.

File_ptr$ is the name of a pointer file that contains two entries. The first entry is the record number and filename containing the **LIFETIME_MOS** record for this particular **HC_LINK_MOS** record. The second entry contains a pointer to a file which in turn points to the parameter data recorded during the test. File_ptr$ always refers to a file in the /var/opt/WLR/database/data/hci/all_hci/lifetime_ptr area. The second entry in the File_ptr$ file always refers to a file in the /var/opt/WLR/database/data/hci/all_hci/stress_ptr area. The entries in this file refer to the PDQ-WLR database recordings, in the order in which they occurred. Each entry consists of a record number and data table file name, usually in the /var/opt/WLR/database/data/measured area. If an algorithm failed, or failed to make a database recording, the entry will consist of "0;*". Refer to the *Database Output* section for details regarding these files.

All database files are in comma-separated ASCII format, which can be queried for further study (the *Database Outputs* section shows how to transport these files into other tools.) For example, if the /var/opt/WLR/database/data/Hc_link_mos/ data were transported to a spreadsheet, then the data can be sorted to locate all "Idsat" lifetime tests performed with the same stress conditions. If there is anomalous data, the pointer file information could be used to located the other device parameters or I-V curve.

Finally, if the algFlags variable "Save HCI Tables" is on (which is the default), the data table information is saved in a spreadsheet-compatible ASCII format.

VT1_HCI Algorithm Example: A simple (and less flexible) HCI example algorithm is VT1_HCI. This algorithm performs a JEDEC compliant (JESD28 or JESD60) hot-carrier test. VT1_HCI performs a logarithmic series of hot-carrier stresses and measures the drain current (linear for N-channel and saturation for P-channel) after each stress cycle using the **HCSTRESS_MOS** and **VT_MOS** algorithm. It then calculates the percent change in the drain current and provides this to **LIFETIME_MOS** to calculate the time to 10% change in the drain current. The data is output by a Plot$ statement into the file "/usr/tmp/VT1_HCI.dta". The stress times can be easily changed in the program line with comments "!Stress times". This algorithm is useful for examining the **HCI_MOS** interface and for performing a simple hot-carrier test. Since the data is not stored in a hierarchical format, data analysis is significantly harder and relies on user knowledge of the test conditions.

Executing Sample HCI Files

Both VT1_HCI.A and ALL_HCI.A are HP BASIC programs stored as ASCII files in the /opt/WLR/newconfig/samples area. These files are provided as samples to use with **HCI_MOS**.

If adopted directly, please remember not to edit the sample files in the /opt/WLR/newconfig area; these reference files can save you the inconvenience of a re-installation if you should wish to start over using the Agilent Technologies distributed code.

HCI_MOS will begin looking for its file (Progfile$) in /opt/WLR/local/ . If you want to develop a "private" version of an algorithm, you may place it in your $HOME/ALG/BASIC area. However, make sure that a file with the same name doesn't already exist in /opt/WLR/local, because that area is scanned before $HOME/ALG/BASIC. With this scheme, you can develop codes in your own area, and then make them available for public use by copying them into /opt/WLR/local.

The HCI file must also be a LIF file. To create a LIF version of your file, first GET it, then RE-STORE it using a unique name. ("VT1_HCI" is the recommended name for re-storing VT1_HCI.A).

For example in HP BASIC, you would perform the following for VT1_HCI.A:

1) GET "/opt/WLR/newconfig/samples/VT1_HCI.A"

2) EDIT

3) RE-STORE "/opt/WLR/local/VT1_HCI"

For **HCI_MOS** input, the Progfile$ is "VT1_HCI" and the Alg$ entry is "Vt1_hci".

For the ALL_HCI algorithm, the above steps are not necessary since it is installed ready to use in /opt/WLR/local. For most users, no modification is necessary to ALL_HCI since all changes can be accomplished with the ASCII input file. To run correctly, you must edit and deposit a copy of "all_hci.input" into the /var/opt/WLR/inputs directory. If a number of "all_hci.input" files are developed, then the ALL_HCI algorithm could be modified to read a different input files. For example, an ALL_HCIP algorithm could read "all_hcip.input"

for a p-channel test and an ALL_HCIN could read "all_hcin.input" for an n-channel test. Another method would keep all of the ALL_HCI input files in /var/opt/WLR/inputs, and then copy the one to use to "all_hci.input".

User-Written HCI Algorithms

For a user-written HCI test, you can call any MOS algorithm except **DELTA_M2F** to perform your test. This is because **DELTA_M2F** has an extra set of device variables. **HCI_MOS** could be modified to accommodate these extra device variables. The calling parameters must match the algorithm's parameters exactly including the pin array and the device parameters. The easiest way is to check the algorithm's SUB declaration in the source code.

Test structure

See "Test Structures" in the *Introduction to Hot Carriers* section.

Step-by-Step Action

1) Check if user algorithm alg$ is already loaded. If not, loadsub it.
2) Initialize arrays.
3) Invoke algorithm.
4) Return values.

Error Code Summary

This value is assigned to the output variable if an error occurred during algorithm execution:

Value	Meaning
2.0E+30	Input parameter error.
+XE+32	Operating system error "X" occurred during execution of algorithm.

Compatibility with Previous Versions

This algorithm is compatible with previous HP BASIC test programs.

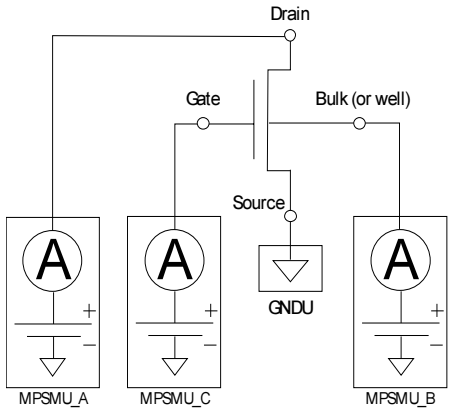
HCSTRESS_MOS

Note: **HCSTRESS_MOS** is obsolete and replaced by **STRSMEAS_2_MOS** in most case. This documentation is for migration and compatibility purposes.

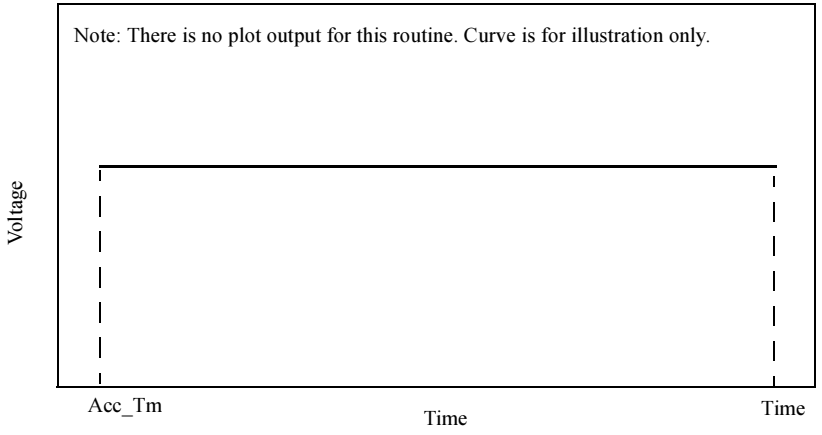
Apply gate, drain and bulk voltage to transistor for hot-carrier-induced degradation stress in compliance with JEDEC standards JESD28 and JESD60 for hot carrier tests.

HCSTRESS_MOS algorithm applies user supplied stress voltages to a **MOSFET** for a specified period of time to induce hot-carrier damage to the device. It is meant to be used in combination with **VT_MOS**, **CALC_VT_MOS** and other algorithms.

Test Configuration



Electrical Characteristics



Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Time	Real	Secs	1.0	0.1~1E6	Time of stress
Vgs†	Real	Volts	2	-100~100	Gate-to-Source stress voltage
Vds†	Real	Volts	5.5	-100~100	Drain-to-Source stress voltage
Vbs†	Real	Volts	0	-100~100	Bulk-to-Source stress voltage
Acc_tm	Real	Secs	0	0~1.0E6	Previous accumulated stress time

† Values are for an N-channel MOSFET. Typically, Vds for a P-channel MOSFET would be less than zero,
Vbs would be greater than zero and Vgs would be less than zero.

Device Variables

Var Name	Var Type	Comment/Description
Width	Real	um, Mask channel width
Length	Real	um, Mask channel length
Type$	String	(N/P), Channel type

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Tstress	Real	Secs	Accumulated Time

Algorithm Syntax

The syntax for calling the algorithm directly (that is, outside of Agilent SPECS) is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction to Hot Carriers* section.

Hcstress_mos (Time, Vgs, Vds, Vbs, Acc_tm,
Pins(*), Width, Length, Type$,
Tstress)

Theory of Operation

The *Introduction to Hot Carriers* section provides a general overview to hot-carrier induced degradation in MOSFETs.

Algorithm **HCSTRESS_MOS** applies a constant user-specified voltage stress at the gate, drain, and/or bulk terminals of a MOSFET for the user-specified time. This algorithm supplies the stress component of the stress/measure cycle described in the *Introduction to Hot Carriers* section. After the stress, transistor parameters are measured and compared to the initial unstressed values. These parameters can include but are not limited to the linear drain current (Ids), maximum transconductance (Gm), threshold voltage (Vtci), punchthrough voltage (Vpt), subthreshold slope (S) as well as charge pumping current (Icp).

The amount of hot-carrier-induced degradation is exponentially dependent on the drain voltage and increasing the drain voltage during stress will reduce test time. However, the maximum drain voltage is determined by the technology and test structure design. A "rule of thumb" limits the initial substrate current (Ibs) to less than 20 microamperes per micron of device width which can be determined using the **MAXSTRS_MOS** algorithm. Another rule limits the ratio of initial substrate current to drain current to less than 15% which can be determined with the **IDIBIG_MOS** algorithm. The hot-carrier-induced degradation also strongly correlates to the gate voltage during stress. An initial value of Vgs can be determined using the **MAXSTRS_MOS** algorithm.

For N-channel MOSFETs in a digital process, parameters typically monitored during stress are Ids, Gm, Vtci, S, and Icp. Stress will cause the |Vtci |, |Icp |, and |S| to increase while |Ids| and |Gm| will decrease. For lightly doped drain (LDD) devices, the hot-carrier-induced threshold voltage shift is usually negligible, and the Gm or Ids change dominates the degradation. For P-channel MOSFET devices, |Vpt | decreases. For devices above 0.5 µm channel length, |Ids| and |Gm| increase while for devices around 0.25 µm, they decrease, and |Vtci | may increase or decrease. Vpt sometimes exhibits the most decrease during P-channel MOSFET hot-carrier-induced degradation.

Test Structure

See "Test Structures" in the *Introduction to Hot Carriers* section.

Step-by-Step Action

1) Check for legal input values. If any are out of range, output variables are set to flag values 2.0E+30 and algorithm exits.
2) Connect SMUs as in Test Configuration diagram in order specified by JESD28 and JESD60.
3) Calculate Stress_time = Time - Acc_tm
4) Force stress voltages.
5) Wait for Stress_time seconds.
6) Return accumulated stress time, Tstress = Time.

Error Code Summary

This value is assigned to the output variable if an error occurred during algorithm execution:

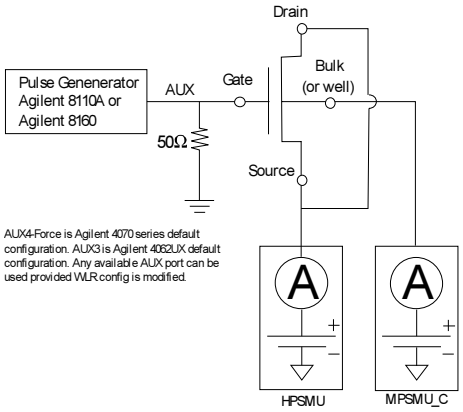
Value	Meaning
2.0E+30	Input parameter error.
1.5E+30	Analysis not performed
+XE+32	Operating system error "X" occurred during execution of algorithm.

CPV_3_MOS

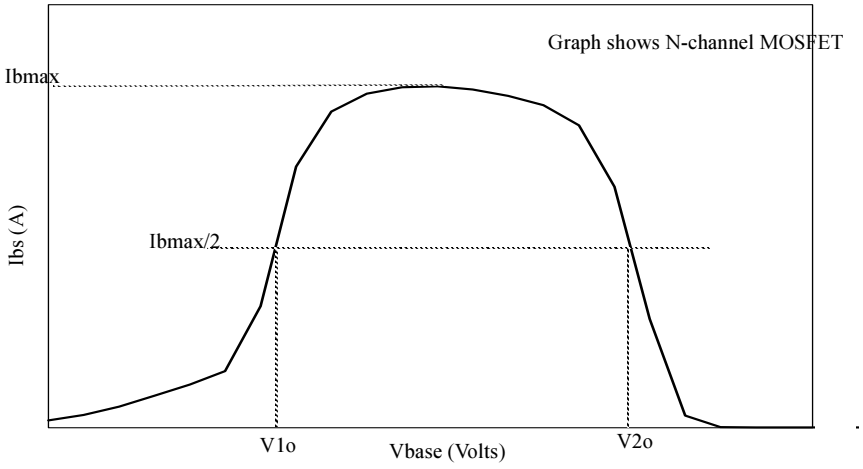
Pulses gate of MOSFET transistor with a trapezoidal waveform at increasing base voltage while measuring substrate current. This routine provides a direct measurement of interface states and an indication of electron and hole trapping in MOS transistors.

CPV_3_MOS test should be performed with the DUT shielded from light. An isolated, floating chuck should be used for measurement. A 50 ohm feed-through termination is required at the AUX connector. The test configuration requires a pulse generator. See *System Configuration* in the *Installation* chapter.

Test Configuration



Electrical Characteristics



Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Vr†	Real	Volts	0	-100~100	Drain and Source bias (Vr = Vd = Vs)
Vb†	Real	Volts	0	-100~100	Bulk bias
P_amp	Real	Volts	3	0.1~9.99	Amplitude of waveform
Freq	Real	Hertz	1E6	1E3~5E6	Frequency of waveform
T_rise	Real	nano-secs	100	20~10E3	Rise time
T_fall	Real	nano-secs	100	20~10E3	Fall time
V_start†	Real	Volts	-4	-9.99~9.89	Vbase start voltage
V_stop†	Real	Volts	1	-9.89~9.99	Vbase stop voltage
Steps	Integer	——	25	1~1001	Number of sweep steps
Integ	Integer	——	2	1~3	SMU integration level 1: Short, 2: Medium, 3: Long
Plot$	String	——	"Y"††		Plot/Save data

† Values are for an N-channel MOSFET. Typically, Vr for a P-channel MOSFET would be less than (or equal to) zero, Vb would be greater than (or equal to) zero.

††A "Y" plots charge pumping current versus base voltage on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files*.

Device Variables

Var Name	Var Type	Comment/Description
Width	Real	um, Mask channel width
Length	Real	um, Mask channel length
Type$	String	(N/P), Channel type

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Ibmax	Real	Amps	Maximum bulk (charge pumping) current
V1o	Real	Volts	Lowest Vbase at Ibmax/2
V2o	Real	Volts	Highest Vbase at Ibmax/2

Algorithm Syntax

The syntax for calling the algorithm directly (that is, outside of Agilent SPECS) is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test can be found in the *Introduction to Hot Carriers* section.

Cpv_3_mos(Vr, Vb, P_amp, Freq, T_rise, T_fall, V_start, V_stop, Steps, Integ,
Plot$,
Pins(*), Width, Length, Type$,
Ibmax, V1o, V2o)

Theory of Operation

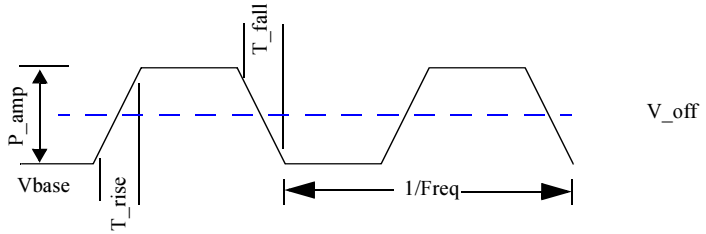
The *Introduction to Hot Carriers* section provides a general overview hot-carrier-induced degradation.

The algorithm **CPV_3_MOS** can be used to characterize the physical mechanisms responsible for hot-carrier-induced degradation of MOSFET devices. Interface-states as well as electron and hole trapping can be observed [1]. The algorithm **CPF_3_MOS** contains additional information on charge pumping.

[1] P. Heremans, J. Witters, G. Groeseneken and H.E. Maes, "Analysis of the charge pumping technique and its application for the evaluation of MOSFET degradation," IEEE Trans. Electron Devices, vol. 36, July 1989, pp. 1318-1335.

Methodology

A trapezoidal waveform is applied to the gate and the DC substrate current is measured (Ib).



The amplitude of the gate waveform (P_amp) remains constant while the base voltage (Vbase) is increased (or decreased) in user specified increments. The amplitude of the pulse (P_amp) should be several times larger than the difference between the threshold (Vtext) and flatband voltage (Vfb). For N-channel MOSFETs, the range of Vbase is chosen so that V_start + P_amp is less than Vfb and V_stop is greater than Vtext. Furthermore, (V_stop + P_amp)/(oxide thickness) or V_start/(oxide thickness) should be less than 6-7 MV/cm so as not to damage the gate oxide. For p-channel MOSFETs, the range of Vbase is chosen so that V_start is less than Vt - P_amp and V_stop is greater than Vfb. Furthermore, (V_start + P_amp)/(oxide thickness) or V_stop/(oxide thickness) should be less than 6-7 MV/cm. A P_amp of 3V (with V_start of -4 V and V_stop of 1 V) is usually adequate for devices in CMOS processes. The user may need to adjust these variables depending on the technology and characterization being done.

Referring to the Electrical Characteristics, the maximum charge pumping current Ibmax (flat part of curve) in the Ibs vs. Vbase curve is proportional to the interface-state density. During a hot-carrier stress, Ibmax will increase dramatically. A shift of the Ibs versus Vbase curve to the left or right can indicate the trapped holes and electrons, respectively. This stretchout is measured by V1o and V2o. These are the lowest and highest voltages at Ibmax/2 and are determined by interpolation.

If the rise and fall time of the gate signal is varied, the interface-state density as function of energy can also be found. The drain voltage (Vd) can be varied to profile the damage as a function of position along the transistor channel.

Testing Notes: For the Agilent 4070 series, the default configuration connects the Agilent 8110A output cable through a 50 Ohm terminator to the AUX4-Force coax. The user may also connect the Agilent 8110A output to a different AUX-Force coax connection (AUX3-AUX6). Do not use an AUX port

which already has a connection to the AUX-Sense port. For example, do not connect to the same AUX port as the Agilent 4140B. If the default AUX port is changed, the user must modify the WLR.config file.

For the Agilent 4062UX, the default configuration connects the Agilent 8110A output cable through a 50 Ohm terminator to the AUX3 port. The user may connect the PGU output to any available AUX port (AUX1-AUX4). If the default AUX port is changed, the user must modify the WLR.config file.

The minimum measurable substrate current is determined by the noise of the test-system wafer chuck. The charge-pumping substrate current may be increased by increasing the waveform frequency or device area. However, the maximum frequency is set by the bandwidth of the test system. For Agilent 4062UX and Agilent 4070 series with DC probe cards, signals with a frequency of 1 MHz and a rise and fall time of 100 ns have been used. The user should verify the integrity of the gate waveform with an oscilloscope. The signal should not have significant "overshoot." The rise and fall times may be reduced to eliminate "overshoot."

The experiment should be performed in the absence of light. The wafer chuck should be isolated and floating.

To speed up measurement of Ibmax, V1o and V2o, the user can:

1) Decrease the number of test points (Steps).

2) Increase the gate pulse frequency (Freq).

To reduce measurement time for production monitoring of interface-states or hot-carrier-induced degradation, the user can reduce Steps to 1. This will perform a Vbase charge pumping experiment and return Ibmax. The user should be set the Vstart and P_amp so that the trapezoidal waveform crosses Vfb and Vtext. In many cases, Icp will be sufficient for monitoring interface-states.

Data File Format: The input parameter Plot$ can be used to save the measured data in an ASCII file using this algorithm. The algorithm saves all the algorithm inputs and outputs into the file first. This is followed by the measured data which is saved in two columns. The first column contains the base voltage of the applied gate pulse while the second column contains corresponding charge pumping current. A row of column headers at the top of the data identifies the parameters stored in individual columns. This format is shown in the following table:

Input parameters	
Output parameters	
Vbase	Icp
<data>	<data>

Test Structure

An N-channel or P-channel MOSFET with separate gate, drain, source, and bulk contacts should be used. Test structures with common gates will not work. The gate should not have any type of input protection (e.g. diode). The transistors should have widths of at least 30 microns. Although smaller width devices will work, larger devices will provide better results. See also "Test Structures" in the *Introduction to Hot Carriers* section.

Step-by-Step Action

1) Check for legal input values. If any are out of range, output variables are set to flag values 2.0E+30 and algorithm exits.
2) Connect SMUs and Pulse Generator as in Test Configuration diagram.
3) Force MOSFET terminal voltages.
4) Pulse gate (trapezoidal waveform of frequency (Freq), amplitude (P_amp), rise time (T_rise) and fall time (T_fall) while increasing (or decreasing) Vbase in increments (V_step) from V_start to V_stop.

   a) Measure Ib three times at each V_base step and average.
5) Search Ib array for maximum bulk current.
6) Determine V1o and V2o at Ibmax/2 by interpolation.

7) Plot/save the data according to the value of Plot$.

8) Return Ibmax, V1o, V2o.

Summary of Error Codes

This value is assigned to the output variable if an error occurred during algorithm execution:

Value	Meaning
1.0E+30	Pulse generator error.
1.5E+30	Analysis not performed
2.0E+30	Input parameter error.
+XE+32	Operating system error "X" occurred during execution of algorithm.

Compatibility with Previous Versions

An algorithm, **CPV_MOS**, is provided to act as a parameter translation shell to maintain compatibility with HP BASIC test programs written for the obsolete **CPV_MOS**. This algorithm, which calls **CPV_3_MOS,** is briefly described next.

CPV_MOS

Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Vr†	Real	Volts	0	-100~100	Drain and Source bias (Vr = Vd = Vs)
Vb†	Real	Volts	0	-100~100	Bulk bias
P_amp	Real	Volts	5	0.1~9.99	Amplitude of waveform
Freq	Real	Hertz	500E3	1E3~5E6	Frequency of waveform
T_rise	Real	nano-secs	200	10~10E3	Rise time
T_fall	Real	nano-secs	200	10~10E3	Fall time
V_start†	Real	Volts	-7	-9.99~9.89	Vbase start voltage
V_stop†	Real	Volts	2	-9.89~9.99	Vbase stop voltage
Steps	Integer	——	101	2~1001	Number of sweep steps
Integ	Integer	——	3	1~3	SMU integration level
Plot$	String	——	"Y"††		Plot/Save data

† Values are for an N-channel MOSFET. Typically, Vr for a P-channel MOSFET would be less than (or equal to) zero, Vb would be greater than (or equal to) zero,

††A "Y" plots charge pumping current versus base voltage on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files*.

Device Variables

Var Name	Var Type	Comment/Description
Width	Real	um, Mask channel width
Length	Real	um, Mask channel length
Type$	String	(N/P), Channel type

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Ibmax	Real	Amps	Maximum bulk current

Algorithm Syntax

The syntax for calling the algorithm directly is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the Introduction section.

Cpv_mos (Vr, Vb, P_amp, Freq, T_rise, T_fall, V_start, V_stop, Steps, Integ, Plot$,
Pins(*), Width, Length, Type$,
Ibmax)

Methodology

CPV_MOS is a translation shell which calls the **CPV_3_MOS** algorithm to conduct the test. The shell maintains compatibility with existing HP BASIC test programs by adding the following output parameters in the call from **CPV_MOS** to **CPV_3_MOS**:

Input parameters:

Not changed.

Output parameters:

V1o: New parameter, not returned.

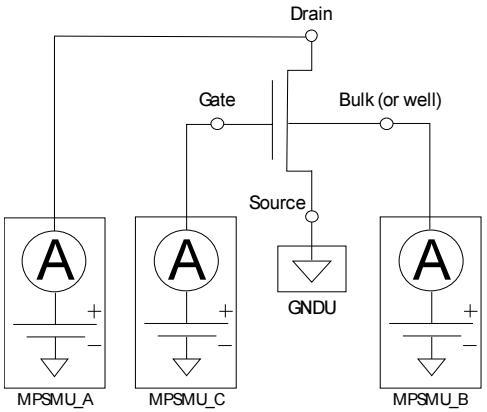
V2o: New parameter, not returned.

VT_MOS

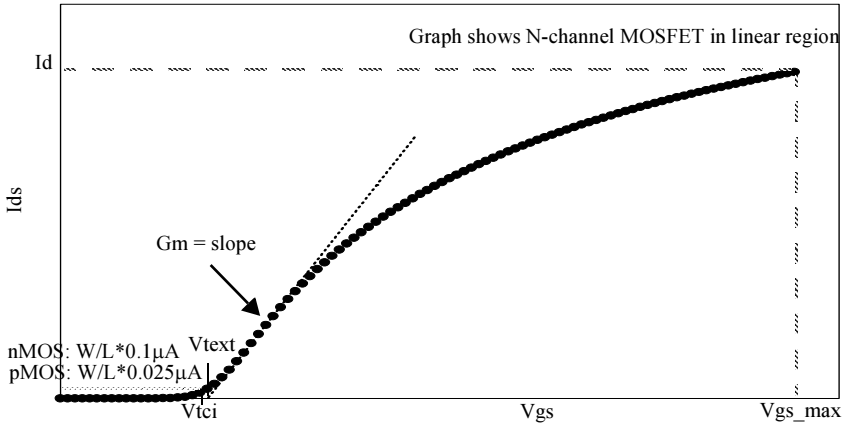
Measures MOSFET threshold voltage, maximum transconductance, and drain currents in compliance with ASTM standards F617-86 and F1096-87, JEDEC standards JESD-28 and JESD-60.

VT_MOS algorithm measures extrapolated threshold voltage (Vtext) constant current threshold voltage (Vtci), maximum transconductance (Gm) and drain current (Id) based on the Id-Vgs curve. For devices operated in the saturation region, the routine can fit $\sqrt{Ids}$ versus Vgs. Measurements can also be made in the forward or reverse directions. This routine is meant to be used for hot-carrier-induced degradation tests in combination with **STRSMEAS_2_MOS**, **HCI_3_MOS** and other algorithms.

Test Configuration



Electrical Characteristics



Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Vds†	Real	Volts	0.1	-100~100	Drain-to-Source bias
Vbs†	Real	Volts	0	-100~100	Bulk-to-Source bias
Vgs_min†	Real	Volts	0	-100~100	Gate-to-Source start bias
Vgs_max†	Real	Volts	3.3	-100~100	Gate-to-Source stop bias
Fr_meas	Integer	——	0	0~1	Measurement direction 0: Forward measurement 1: Reverse measurement
Sq_root	Integer	——	0	0~1	Type of I_{ds} fit 0: Fit I_{ds} 1: Fit square root of I_{ds}
Steps	Integer	——	101	2~1001	Number of sweep steps
Integ	Integer	——	2	1~3	SMU integration level 1: Short, 2: Medium, 3: Long
Plot$	String	——	"Y"††		Plot/Save Data

†Values are for an N-channel MOSFET. Typically, Vds for a P-channel MOSFET would be less than zero,
Vbs would be greater than zero and Vgs would be less than zero.

††A "Y" plots current versus voltage on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files.*

Device Variables

Var Name	Var Type	Comment/Description
Width	Real	um, Mask channel width
Length	Real	um, Mask channel length
Type$	String	(N/P), Channel type

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Vtext	Real	Volts	Extrapolated threshold voltage
Vtci	Real	Volts	Constant current threshold voltage
Gm	Real	µA/V or µA/V^2	Max Transconductance (units µA/V^2 if Sq_root = 1)
Id	Real	Amps	Drain current at Vgs_max

Algorithm Syntax

The syntax for calling the algorithm directly (that is, outside of Agilent SPECS) is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction to Hot Carriers* section.

Vt_mos (Vds, Vbs, Vgs_min, Vgs_max, Fr_meas, Sq_root, Steps, Integ, Plot$,
Pins(*), Width, Length, Type$,
Vtext, Vtci, Gm, Id)

Theory of Operation

VT_MOS is an integral part on any hot-carrier-induced degradation test. The *Introduction to Hot Carriers* section provides a general overview of hot-carrier-induced degradation in transistors.

VT_MOS measures MOSFET threshold voltage, maximum transconductance (Gm) and drain current at Vgs_max (Id). Both a constant current threshold (Vtci) and extrapolated threshold (Vtext) measurement are provided. Gm is the maximum slope of the drain current versus gate voltage curve and is proportional to the device mobility and width while inversely proportional to oxide thickness and device length. Changes in Gm, Vtci, Vtext or Id during a

hot-carrier stress may be due to generated interface states, trapped charge or both. To quantify the damage mechanism, the user is referred to the **CPV_3_MOS** and/or **CPF_3_MOS** algorithms.

These measurements may be made in the forward or reverse mode. A forward measurement is made with the source and drain terminals in the same orientation as during hot-carrier stress. A reverse measurement is made with the source (drain) terminal switched with the drain (source) stress terminals.

Forward and reverse measurements are useful for characterizing hot-carrier-induced degradation. Since hot-carrier-induced damage is localized near the drain junction, a reverse measurement will show more degradation than a forward measurement (particularly in saturation). These measurements can be used to check that a hot-carrier stress is proceeding properly.

Methodology

Vtext and Gm are determined by a linear least-squares fit to the Id-Vgs data as described in ASTM standard F616-86, JEDEC standards JESD28 and JESD60. Vtci defined as the gate voltage required for the following drain currents:

$$I_d = 0.1\mu A \cdot \frac{W}{L}$$ N-channel MOSFET (per JESD28)

$$I_d = 0.025\mu A \cdot \frac{W}{L}$$ P-channel MOSFET (per JESD60)

where W = transistor width (μm)
L = transistor length (μm).

Vtci is determined by an interpolation of the Id-Vgs data as recommended in JESD28 and JESD60. The step size (Steps) will influence the resolution of the measurements. This resolution determines the minimum measurable percent change during a hot-carrier stress.

Id is the drain current at the specified Vds and Vgs_max. Typical Vgs_max in JESD28 and JESD60 are defined as Vdd. The drain current may also be measured with the **IDIBIG_MOS** algorithm.

Testing Notes: For saturation measurement with submicron transistors, it is recommended that you use Sq_root=1. This will usually give the best results.

Data File Format: The input parameter Plot$ can be used to save the measured data in an ASCII file using this algorithm. The algorithm saves all the algorithm inputs and outputs into the file first. This is followed by the measured data saved in two columns. The first column contains the gate biases while the second column contains the drain currents measured at the corresponding gate biases. A row of column headers at the top of the data identifies the parameters stored in individual columns. This format is shown in the following table:

Input parameters	
Output parameters	
V_{gs} (V)	I_{ds} (A)
<data>	<data>

Test Structure

Requires a N-channel or P-channel MOSFET. Additional information is in "Test Structures" in the *Introduction to Hot Carriers* section.

Step-by-Step Action

1) Check for legal input values. If any are out of range, output variables are set to flag values 2.0E+30 and algorithm exits.
2) Connect SMU's as in Test Configuration diagram.
   a) Swap drain and source pins if a "reverse" measurement is required.
3) Set SMU integration to Integ input value.
4) Force MOSFET terminal voltages (Vbs, Vds).
5) Sweep Vgs voltage from Vgs_min to Vgs_max in increments of Steps.
6) Measure Ids at each step and build Vgs vs. Ids arrays.
7) Take absolute values of Ids array and use to calculate Vtci, Vtext, Gm and Id.
   a) If device open or short based on Vtci target current, set outputs to 1E+36 or 1E-99, respectively. Exit algorithm.
   b) Extract Id from Ids array.
   c) Determine Vtci by interpolation.
   d) Find maximum slope by looping through data between Vgs_max and Vgs_min points.
   e) Use this slope to calculate Vtext.
   f) Set Gm equal to absolute value of maximum slope.
8) Plot/save the data according to the value of Plot$.
9) Return Vtext, Vtci, Gm and Id.

Error Code Summary

This value is assigned to the output variable if an error occurred during algorithm execution:

Value	Meaning
1.0E-99	DUT is assumed to be shorted
1.0E+36	DUT is assumed to be open
2.0E+30	Input parameter error
1.5E+30	Analysis not performed
+XE+32	Operating system error "X" occurred during execution of algorithm

Compatibility with Previous Versions

This algorithm is compatible with previous HP BASIC test programs.

Please note that the definition for Vtci has changed from the previous release to be compatible with JEDEC standards JESD28 and JESD60. Previously, Vtci was defined to be 1 μA*W/L for both N-channel and P-channel MOSFETs. Now, the target current for Vtci is 0.1 μA*W/L for N-channel devices and 0.025 μA*W/L for P-channel devices.

Also note that the algorithm output value of an open DUT has been changed from 1E+99 to 1E+36. This is done to maintain compatibility with Agilent SPECS maximum input value.

CALC_VT_MOS

Determine absolute and percent change in Vtext, Vtci, Gm and Id from initial values during a hot-carrier stress.

CALC_VT_MOS algorithm determines absolute and percent change in Vtext, Vtci, Gm and Id from initial values. It is designed to be used in conjunction with the **VT_MOS** algorithm and **HCSTRESS_MOS** during a hot-carrier stress.

There is no device curve to plot or test configuration needed.

Input Variables

Var Name	Var Type	Var Unit	Default Value	Range[1] (min~max)	Comment/Description
Vtext†	Real	Volts	1	-1.0E36~1.0E36	Extrapolated threshold voltage at Time = 0 seconds
Vtci†	Real	Volts	1	-1.0E36~1.0E36	Constant I threshold voltage at Time = 0 seconds
Gm	Real	µA/V or µA/V^2	1	1.0E-99~1.0E36	Maximum transconductance at Time = 0 seconds
Id	Real	Amps	1	1.0E-99~1.0E36	Drain current at Time = 0 seconds
Vtext_p	Real	Volts	1	-1.0E36~1.0E36	Extrapolated threshold voltage at Time = Stress
Vtci_p	Real	Volts	1	-1.0E36~1.0E36	Constant current threshold voltage at Time = Stress
Gm_p	Real	µA/V or µA/V^2	1	1.0E-99~1.0E36	Maximum transconductance at Time = Stress
Id_p	Real	Amps	1	1.0E-99~1.0E36	Drain current at Time = Stress

† Values of zero are set to 1.0E-99 to avoid division by zero

1. The allowed limits have been reduced to the maximum input value for Agilent SPECS.

Device Variables

Var Name	Var Type	Comment/Description
Width	Real	um, Mask channel width
Length	Real	um, Mask channel length
Type$	String	(N/P), Channel type

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Dvtext	Real	Volts	Change in Vtext
Pvtext	Real	%	Percent change in Vtext
Dvtci	Real	Volts	Change in Vtci
Pvtci	Real	%	Percent change in Vtci
Dgm	Real	mA/V or mA/V^2	Change in Gm
Pgm	Real	%	Percent change in Gm
Did	Real	Amps	Change in Id
Pid	Real	%	Percent change in Id

Algorithm Syntax

The syntax for calling the algorithm directly (that is, outside of Agilent SPECS) is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction to Hot Carriers* section.

Calc_vt_mos (Vtext, Vtci, Gm, Id, Vtext_p, Vtci_p, Gm_p, Id_p,
Pins(*), Width, Length, Type$,
Dvtext, Pvtext, Dvtci, Pvtci, Dgm, Pgm, Did, Pid)

Theory of Operation

This routine calculates the difference and percent difference from the initial, Time = 0, values and the stress values of transistor parameters. These values are routinely used for monitoring hot-carrier-induced degradation of MOSFETs.

Test structure

Not applicable (see **STRSMEAS_2_MOS** or **VT_MOS**).

Step-by-Step Action

1) Check for legal input values. If any are out of range, output variables are set to flag values 2.0E+30 and algorithm exits.

   a) Set initial threshold voltages equal to zero to 1.0E-99.

2) Calculate difference and percent difference in Vtext, Vtci, Gm and Id.

3) Return values for Dvtext, Pvtext, Dvtci, Pvtci, Dgm, Pgm, Did, Pid.

Summary of Error Codes

This value is assigned to the output variable if an error occurred during algorithm execution:

Value	Meaning
2.0E+30	Input parameter error.
1.5E+30	Analysis not performed
+XE+32	Operating system error "X" occurred during execution of algorithm.

Compatibility with Previous Versions

This algorithm is compatible with previous HP BASIC test programs.

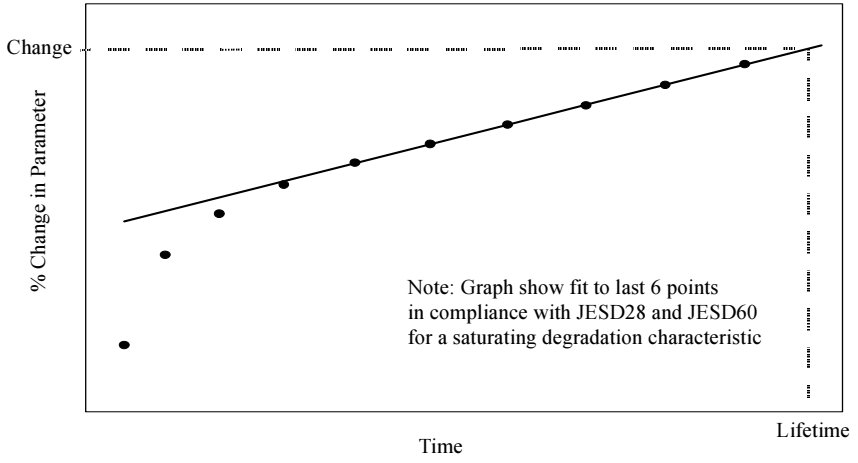
The range of the maximum allowed input variables has been reduced from 1E+99 to 1E+36 to maintain compatibility with Agilent SPECS.

LIFETIME_2_MOS

Calculate the time for a parameter to degrade to the user specified condition for hot-carrier-induced degradation of MOSFETs in compliance with JEDEC standards JESD28 and JESD60.

Given input arrays representing a parameter over time, **LIFETIME_2_MOS** will extrapolate or interpolate to the point in time when this parameter will degrade to the user specified condition. The calculation is designed for use with hot-carrier-induced degradation algorithms.

Electrical Characteristics



Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Pname$	String	——	"--"	" "	Parameter name for Y axis label
Pa†	Real Array	——	——	0~1E36	Parameter array (1000x1)
Ti†	Real Array	Secs	——		Time array (1000x1)
Change	Real	——	10	0.01~1E6	Change at lifetime
Units$	String	——	" "	" "	Units: "Percent" or EU (Engineering Units)
EU	Integer	——	0	0~1	EU Flag: 1=EU, 0=%
Npts	Integer	——	20	2~1000	Number of points in array
Lpts	Integer	——	8	2~1000	Number points in fit starting with last/first data point
X_axis	Integer	——	1	0~1	Axis type. 0: linear 1:log
Y_axis	Integer	——	1	0~1	Axis type. 0: linear 1:log
Plot$	String	——	"Y"††		Plot/Save data

† Array elements must be specified as absolute value.

††A "Y" plots parameter versus time on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files*.

Device Variables

Var Name	Var Type	Comment/Description
Width	Real	um, Mask channel width
Length	Real	um, Mask channel length
Type$	String	(N/P), Channel type

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Lifetime	Real	Secs	Estimated time to Change
Corcoef	Real	——	Correlation coefficient
Slope	Real	——	Slope of Parameter vs Time
Intcpt	Real	——	Intercept Parameter vs Time

Algorithm Syntax

The syntax for calling the algorithm directly (that is, outside of Agilent SPECS) is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction to Hot Carriers* section.

LIFETIME_2_MOS(Pname$, Pa(*), Ti(*), Change, Units$, EU, Npts, Lpts,
X_axis, Y_axis, Plot$,
Pins(*)†, Width, Length, Type$,
Lifetime, Corcoef, Slope, Intcpt)

Theory of Operation

The *Introduction to Hot Carriers* section provides a general overview of hot-carrier-induced degradation.

LIFETIME_2_MOS is used to estimate the time-to-specified degradation level resulting from hot-carrier-induced degradation. Parameters such as linear drain current (Idslin), saturated drain current (Idsat), maximum transconductance (Gmlin) or threshold voltage (Vtci or Vtext) are measured during a hot-carrier stress as a function of cumulative stress time. These are used to extrapolate or interpolate the time to a specified amount of degradation. The time-to-specified degradation is sometimes called the "lifetime". This "lifetime" is not the lifetime of an IC. Rather, it is a convenient

measure of the damage caused by hot carriers. It can be used to compare the hot-carrier immunity of various processes. Time-to-10% degradation in Idslin, Gm, or Vt are sometimes used as "lifetime" metrics for fast tests. In the past, the time to 20 mV or 100 mV change in threshold voltage has also been used. The percent change in Idslin is calculated as [Idslin(t) - Idslin(0)] / Idslin(0)). The absolute change in threshold voltage is Vtci(t) - Vtci(0). The time-to-5% degradation in forward saturation drain current (Idsat) is also sometimes used as a failure criterion for digital circuits.

The hot-carrier-induced degradation of N-channel MOSFET devices typically follow a power-law (with time). That is, the degradation on a logarithmic scale versus the time on a logarithmic scale is a straight line. For devices with lightly doped drains (LDDs), the degradation curve can saturate as shown in the Electrical Characteristic example. This saturation can be taken into account by fitting the last points (Lpts) of the curve. A conservative estimate of the "lifetime" can also be made by fitting all the data points. The hot-carrier-induced degradation of P-channel MOSFETs may have a logarithmic time dependence (particaullarly for devices stressed at peak gate current in technologies with minimum gate lengths above 0.25 um). That is, the degradation versus the logarithm of time is a straight line. A fit to this type of data can also be performed with this routine.

Methodology

LIFETIME_2_MOS first determines if interpolation or extrapolation is needed to estimate Lifetime. If the minimum Y-value is below Change and the maximum Y-value is above Change, then a 2-point Lagrange interpolation is used to calculate Lifetime. Otherwise extrapolation is used. If the largest Y-data point is less than Change, then extrapolation forward is used. Here, the last Lpts of data are fit to a linear, logarithmic, exponential, or power law depending on the axis-types as shown below:

X axis	Y axis	Fit Type	Equation
0 : linear	0 : linear	linear	y = Intcpt + Slope * x
1 : logarithmic	0 : linear	logarithmic	y = Intcpt + Slope * $\log_{10}(x)$
0 : linear	1 : logarithmic	exponential	$\log_{10}(y)$ = Intcpt + Slope * x
1 : logarithmic	1 : logarithmic	power	$\log_{10}(y)$ = Intcpt + Slope * $\log_{10}(x)$

If Change is greater than the largest Y-data point, then extrapolation backward is used. Here, the first Lpts of data are fit in accordance with the axis types.

If the correlation coefficient (Corcoef) from the fit is less than 0.5, we consider this data to have no trend and the extrapolation of lifetime to be meaningless. In this case, the Lifetime is returned as Missing.

The models in **LIFETIME_2_MOS** assume a single dominant degradation mechanism. As a result we assume that the slope of |% parameter| or |delta parameter| should be increasing (positive). If the calculated slope is less than 0, then the result is not consistent with a single degradation mode and the lifetime is returned as Missing. In most cases, a negative slope results from a fit to measurement noise. In this case, the stress time should be increased or the stress level increased.

Data File Format: The input parameter Plot$ can be used to save the analyzed data in an ASCII file using this algorithm. The algorithm saves all the algorithm inputs and outputs into the file first. This is followed by the data which is saved in two columns. The first column contains the stress time while the second column contains the parameter values at the corresponding stress times. A row of column headers at the top of the data identifies the parameters stored in individual columns. This format is shown in the following table:

Input parameters	
Output parameters	
Time (sec)	Parameter
<data>	<data>

Test Structure

No structure is necessary for this routine.

Step-by-Step Action

1) Check for legal input values. If any are out of range, output variables are set to flag values 2.0E+30 and algorithm exits.

2) Resize internal arrays to Npts size, enter input arrays and format according to X_axis and Y_axis (i.e. linear or log).

   a) If X or Y data = 0 and Logarithmic axis, Set point(s) to 1E-6 for % or 1E-15 for E.U.

3) Determine if interpolation or extrapolation is required:

   a) If interpolation, find Change point using linear interpolation (set Corcoef, Slope, and Intcpt to value of variable Missing).

   b) If extrapolation (forward), least squares fit to last Lpts

   c) If extrapolation (backward), least square fit to first Lpts

   d) If Corcoef < 0.5, Lifetime set to -Missing, else calculate Lifetime

   e) If slope < 0, Lifetime set to -Missing, else calculate Lifetime.

   f) If Lifetime <= 0, Lifetime set to -Missing.

4) Plot/save the data according to the value of Plot$.

5) Return Lifetime, Corcoef, Slope, and Intcpt.

Error Code Summary

This value is assigned to the output variable if an error occurred during algorithm execution:

Value	Meaning
2.0E+30	One or more input parameters are out of range.
+/- 1.5E+30	Analysis not performed
+XE+32	Operating system error "X" occurred during execution of algorithm.

Compatibility with Previous Versions

LIFETIME_MOS is provided to act as a parameter translatioin shell to maintain compatibility with HP BASIC test programs written for the obsolete **LIFETIME_MOS**. This algorithm, which calls **LIFETIME_2_MOS** is briefly described below.

LIFETIME_MOS

Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Pname$	String	——	"--"	" "~" "	Parameter name for Y axis label
Pa†	Real Array	——	——	0~1E36	Parameter array (1000x1)
Ti†	Real Array	Secs	——		Time array (1000x1)
Change	Real	——	10	0.01~1E6	Change at lifetime
Npts	Integer	——	20	2~1000	Number of points in array
Lpts	Integer	——	8	2~1000	Number points in fit starting with last/first data point
X_axis	Integer	——	1	0~1	Axis type. 0: linear 1:log
Y_axis	Integer	——	1	0~1	Axis type. 0: linear 1:log
Plot$	String	——	"Y"††		Plot/Save data

† Array elements must be specified as absolute value.

††A "Y" plots parameter versus time on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files*.

Device Variables

Var Name	Var Type	Comment/Description
Width	Real	um, Mask channel width
Length	Real	um, Mask channel length
Type$	String	(N/P), Channel type

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Lifetime	Real	Secs	Estimated time to Change
Corcoef	Real	——	Correlation coefficient
Slope	Real	——	Slope of Parameter vs Time
Intcpt	Real	——	Intercept Parameter vs Time

Algorithm Syntax

The syntax for calling the algorithm directly (that is, outside of Agilent SPECS) is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction to Hot Carriers* section.

Lifetime_mos (Pname$, Pa(*), Ti(*), Change, Npts, Lpts, X_axis, Y_axis, Plot$,
Pins(*)†, Width, Length, Type$,
Lifetime, Corcoef, Slope, Intcpt)

Methodology

The algorithm is a translation shell which calls the **LIFETIME_2_MOS** algorithm to conduct the calculation. The shell maintains compatiblity with existing HP BASIC test programs by adding the following input parameters in the call from **LIFETIME_MOS** to **LIFETIME_2_MOS**.

Input Parameters:

Units$: Added and set to "Percent".

EU: Added and set to 0 to do percent.

Output Parameters:

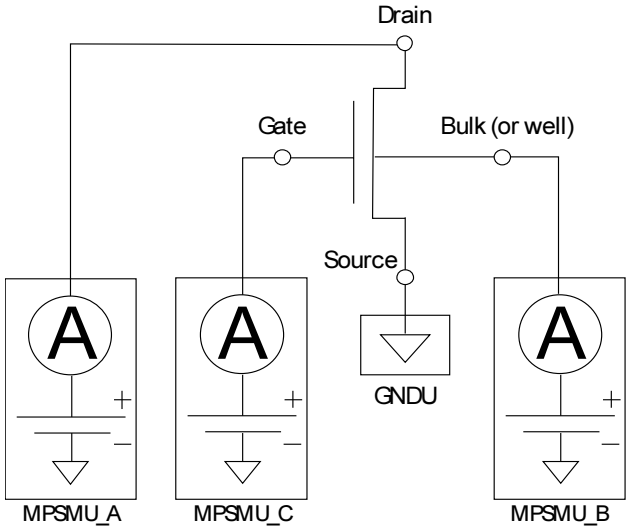
No Changes

IDIBIG_MOS

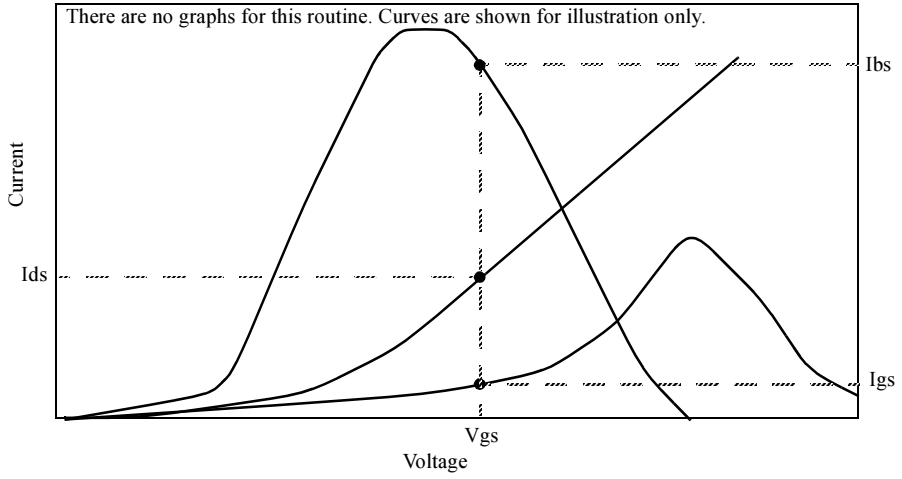
Measure MOSFET drain, bulk, and gate currents at user specified Vgs and Vds conditions.

There is no device curve to plot.

Test Configuration



Electrical Characteristics



Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Vds†	Real	Volts	5.5	-100~100	Drain-to-Source bias
Vgs†	Real	Volts	2.2	-100~100	Gate-to-Source bias
Vbs†	Real	Volts	0	-100~100	Bulk-to-Source bias
Fr_meas	Integer	——	0	0~1	Measurement direction 0: Forward measurement 1: Reverse measurement
Integ	Integer	——	2	1~3	SMU integration level 1: Short, 2: Medium, 3: Long

† Values are for an N-channel MOSFET. Typically, for a P-channel MOSFET, Vds would be less than zero,
Vbs would be greater than zero, and Vgs would be less than zero.

Device Variables

Var Name	Var Type	Comment/Description
Width	Real	um, Mask channel width
Length	Real	um, Mask channel length
Type$	String	(N/P), Channel type

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Ids	Real	Amps	Drain current
Ibs	Real	Amps	Substrate current
Igs	Real	Amps	Gate current

Algorithm Syntax

The syntax for calling the algorithm directly (that is, outside of Agilent SPECS) is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction to Hot Carriers* section.

Idibig_mos (Vds, Vgs, Vbs, Fr_meas, Integ,
Pins(*), Width, Length, Type$,
Ids, Ibs, Igs)

Theory of Operation

The *Introduction to Hot Carriers* section provides a general overview of hot carrier degradation.

IDIBIG_MOS measures the drain, substrate and gate current of a device at user specified drain (Vds), gate (Vgs), and bulk (Vbs) voltages. This algorithm is useful for measuring gate (Igs), substrate (Ibs), and drain (Ids) currents before or during a hot-carrier stress. The initial values are routinely used to scale the hot-carrier-induced "lifetime" at operating conditions from accelerated stresses. Furthermore, Ibs/Ids or Ibs can be used to set the maximum drain voltage for a hot-carrier stress (see **STRSMEAS_2_MOS**).

These measurements may be made in the forward or reverse mode. A forward measurement is made with the source and drain terminals in the same orientation as during the hot-carrier stress. A reverse measurement is made with the source (drain) terminal switched with the drain (source) stress terminals.

This routine can also be used to measure the linear drain current (e.g., current with Vds=0.1 V and Vgs=3.3 V) or saturation current (e.g., Vds=Vgs=3.3 V) during a hot-carrier stress. In this mode, this routine is meant to be used with **CALC_ID_MOS** to determine the change in these parameters due to hot-carrier damage. These drain currents measurements can also be made with the **VT_MOS** algorithm.

Test structure

See "Test Structures" in the *Introduction to Hot Carriers* section.

Step-by-Step Action

1) Check for legal input values. If any are out of range, output variables are set to flag values 2.0E+30 and algorithm exits.
2) Connect SMU's as in Test Configuration diagram.

   a) Swap drain and source pins if a "reverse" measurement is required.
3) Set SMU integration time to Integ input value.
4) Force MOSFET terminal voltages (Vbs, Vgs, Vds) in order specified by JESD28 and JESD60.
5) Measure Ids, Ibs, and Igs.
6) Return absolute values for Ids, Ibs, and Igs.

Error Code Summary

This value is assigned to the output variable if an error occurred during algorithm execution:

Value	Meaning
2.0E+30	Input parameter error.
1.5E+30	Analysis not performed
+XE+32	Operating system error "X" occurred during execution of algorithm.

Compatibility with Previous Versions

This algorithm is compatible with previous HP BASIC test programs.

CALC_ID_MOS

Calculate absolute and percent change in currents from initial values during a hot carrier stress.

The **CALC_ID_MOS** algorithm determines absolute and percent change in drain, source, and bulk currents from initial values. It is designed to be used in conjunction with the **IDIBIG_MOS** and **HCSTRESS_MOS** algorithms during a hot-carrier stress.

There is no device curve to plot or test configuration needed.

Input Variables

Var Name	Var Type	Var Unit	Default Value	Range[1] (min~max)	Comment/Description
Ids	Real	Amps	1	1.0E-99~1.0E36	Drain current at Time = 0 seconds
Ibs	Real	Amps	1	1.0E-99~1.0E36	Bulk current at Time = 0 seconds
Igs	Real	Amps	1	1.0E-99~1.0E36	Gate current at Time = 0 seconds
Ids_p	Real	Amps	1	1.0E-99~1.0E36	Drain current at Time = Tstress
Ibs_p	Real	Amps	1	1.0E-99~1.0E36	Bulk current at Time =Tstress
Igs_p	Real	Amps	1	1.0E-99~1.0E36	Gate current at Time = Tstress

1. Maximum value have been reduced from 1E+99 to 1E36 to be compatible with Agilent SPECS maximum input value.

Device Variables

Var Name	Var Type	Comment/Description
Width	Real	um, Mask channel width
Length	Real	um, Mask channel length
Type$	String	(N/P), Channel type

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Dids	Real	Amps	Change in Ids
Pids	Real	%	Percent change in Ids
Dibs	Real	Amps	Change in Ibs
Pibs	Real	%	Percent change in Ibs
Digs	Real	Amps	Change in Igs
Pigs	Real	%	Percent change in Igs

Algorithm Syntax

The syntax for calling the algorithm directly (that is, outside of Agilent SPECS) is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction to Hot Carriers* section.

Calc_id_mos (Ids, Ibs, Igs, Ids_p, Ibs_p, Igs_p,
Pins(*), Width, Length, Type$,
Dids, Pids, Dibs, Pibs, Digs, Pigs)

Theory of Operation

This routine calculates the difference and percent difference from the initial, (Time = 0), values and the stress values of transistor parameters.

Test structure

Not applicable.

Step-by-Step Action

1) Check for legal input values. If any are out of range, output variables are set to flag values 2.0E+30 and algorithm exits.

2) Calculate difference and percent difference in Ids, Ibs and Igs.

3) Return values for Dids, Pids, Dibs, Pibs, Digs, Pigs.

Summary of Error Codes

This value is assigned to the output variable if an error occurred during algorithm execution:

Value	Meaning
2.0E+30	Input parameter error.
1.5E+30	Analysis not performed
+XE+32	Operating system error "X" occurred during execution of algorithm.

Compatibility with Previous Versions

This algorithm is compatible with previous HP BASIC test programs.

The range of the maximum allowed input variables has been reduced from 1E+99 to 1E+36 to maintain compatibility with Agilent SPECS.

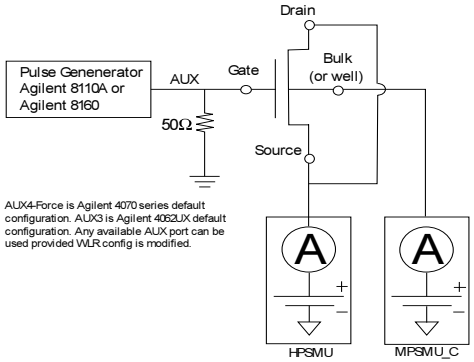
CPF_3_MOS

Measure average interface-state density (Dit) of MOSFET device with charge pumping in compliance with ASTM Standard F1340-91.

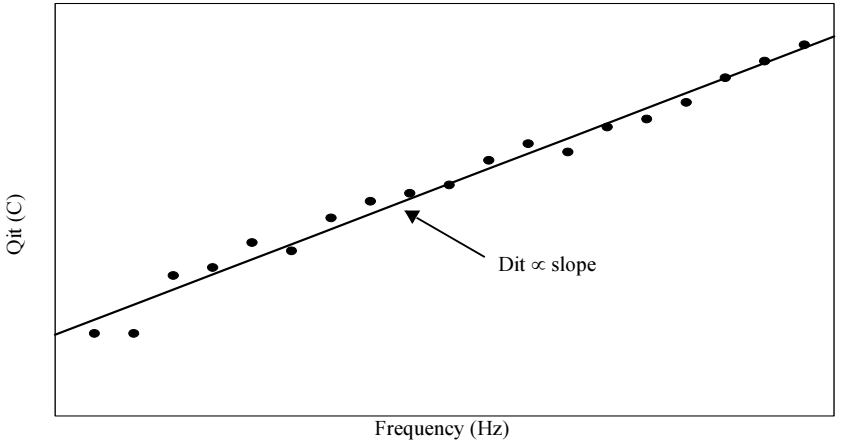
CPF_3_MOS should be performed with the DUT shielded from light. An isolated floating chuck should be used for the measurement. A 50 ohm feed-through termination is required at the AUX connector (if possible, termination at the probe card is preferred).

The test configuration requires a pulse generator. See *System Configuration* in the *Installation* chapter.

Test Configuration



Electrical Characteristics



Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Leff	Real	µm	0.3	0~1000	Effective device length
Weff	Real	µm	30	0~10E3	Effective device width
Vr†	Real	Volts	0	-100~100	Drain and Source bias (Vr=Vd=Vs)
Vb†	Real	Volts	0	-100~100	Bulk bias
Icp_min	Real	pA	10	1E-3~1E3	Minimum measurable charge pumping current
P_amp	Real	Volts	6	0.1~9.99	Peak-to-peak amplitude of gate signal
V_off	Real	Volts	0	-8.55~9.40	Offset voltage, center of gate signal
F_min	Real	Hz	100E3	10~10E6	Minimum Frequency
F_max	Real	Hz	5E6	10~10E6	Maximum Frequency
Nf	Integer	——	15	1~70	Number of frequencies
Integ	Integer	——	3	1~3	SMU integration level 1: Short, 2: Medium, 3: Long
Plot$	String	——	"Y"††		Plot/Save data

† Values are for an N-channel MOSFET. Typically, Vr for a P-channel MOSFET would be less than (or equal to) zero, Vb would be greater than (or equal to) zero.

††A "Y" plots Qit versus frequency on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files.*

Device Variables

Var Name	Var Type	Comment/Description
Width	Real	um, Mask channel width
Length	Real	um, Mask channel length
Type$	String	(N/P), Channel type

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Dit	Real	$cm^{-2}eV^{-1}$	Average interface-state density
Icp	Real	Amps	Charge pump current at F_max
Corcoef	Real		Correlation coefficient of Qit vs. Freq. fit

Algorithm Syntax

The syntax for calling the algorithm directly (that is, outside of Agilent SPECS) is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction to Hot Carriers* section.

Cpf_mos (Leff, Weff, Vr, Vb, Icp_min, Vph, Vpl, F_min, F_max, Nf, Integ,
Plot$,
Pins(*), Width, Length, Type$,
Dit, Icp, Corcoef)

Theory of Operation

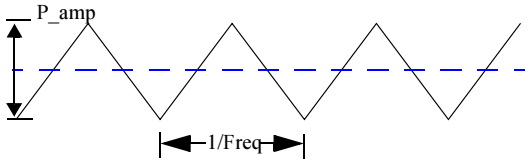
The *Introduction to Hot Carriers* section provides a general overview of hot-carrier-induced degradation of transistors.

CPF_3_MOS uses charge pumping to calculate the average interface-state density in a MOS transistor [1]. The **CPV_3_MOS** also contains additional information on charge pumping. The interface-state density is a fundamental MOSFET device parameter. Unlike the algorithm **CPV_3_MOS**, this routine varies the frequency of a fixed-voltage sawtooth waveform. This algorithm is useful for routine monitoring of hot-carrier-induced degradation. The advantage **CPV_3_MOS** over **HFCV_3_CAP** and **QCV_3_CAP** in measurement of interface state density is that it may be performed on minimum channel length transistors as part of in-line tests, is faster and is 10-100 times more sensitive.

[1] G. Groeseneken, H.E. Maes, N. Beltran and R.F. De Keersmaekcker, "A reliable approach to charge-pumping measurements in MOS transistors," IEEE Trans. Electron Devices, vol. 31, Jan. 84, pp. 42-53.

Methodology

A sawtooth waveform is applied to the gate and DC substrate current (Ib), also called charge pumping current, is measured. The slope of the Electrical Characteristic curve (Icp/frequency versus frequency) is used to calculate the average interface-state density (Dit).



The maximum waveform amplitude (P_amp) should be several times larger than the sum of threshold voltage (Vtext) and the flatband voltage (Vfb) as:

$$P\_amp > 4*(|Vtext|+|Vfb|).$$

The offset voltage (V_off) can be determined in accordance with ASTM F1340-91 or estimated by

$$V\_off = (Vtext + Vfb)/2.$$

For conventional digital CMOS processes, P_amp=6V and V_off=0 should be a good starting point.

The minimum frequency (F_min) is set by the "noise level" (minimum measurable charge pumping current) of the system, Icp_min. The "noise level" will depend on the noise of the wafer chuck. This can be determined by running a charge pumping experiment with the pulser disconnected. The minimum frequency can be decreased by increasing device area (WxL). A larger device produces more charge pumping current. The maximum frequency is set by the bandwidth of the test system.

Testing Notes: For the Agilent 4070 series, the default configuration connects the Agilent 8110A output cable through a 50 Ohm terminator to the AUX4-Force coax. The user may also connect the Agilent 8110A output to a different AUX-Force coax connection (AUX3-AUX6). Do not use an AUX port which already has a connection to the AUX-Sense port. For example, do not connect to the same AUX port as the Agilent 4140B. If the default AUX port is changed, the user must modify the WLR.config file.

For the Agilent 4062UX, the default configuration connects the Agilent 8110A output cable through a 50 Ohm terminator to the AUX3 port. The user may connect the PGU output to any available AUX port (AUX1-AUX4). If the default AUX port is changed, the user must modify the WLR.config file.

The experiment should be performed in the absence of light. The wafer chuck should be isolated and floating.

To speed up measurement of Dit, the user can:

1) Decrease the number of test points (Nf).

2) Change Integ to 2 (medium integration). In this case, it may be necessary to increase Icp_min.

3) Increase the F_min and F_max. This will increase charge pumping current and reduce measurement time.

To reduce measurement time for production monitoring of Dit or hot-carrier-induced degradation, the user can reduce Nf to 1. F_max should be 1 MHz or above. This will perform a single frequency charge pumping experiment and return Icp. In many cases, Icp will be sufficient for monitoring interface-states.

Data File Format: The input parameter Plot$ can be used to save the measured data in an ASCII file using this algorithm. The algorithm saves all the algorithm inputs and outputs into the file first. This is followed by the measured data which is saved in two columns. The first column contains the frequency of the applied gate pulse while the second column contains the corresponding interface trapped charge, Qit=Icp/f, contributing to the charge pumping current per cycle of the gate pulse. A row of column headers at the top of the data identifies the parameters stored in individual columns. This format is shown in the following table:

Input parameters	
Output parameters	
Frequency	Q_{it}
<data>	<data>

Test Structure

An N-channel or P-channel MOSFET with separate gate, drain, source, and bulk contacts should be used. Test structures with common gates will not work. The gate should not have any type of input protection (e.g. diode). The transistors should have widths of at least 30 microns. Although smaller devices will work, larger devices will provide better results. See also "Test Structure" in the *Introduction to Hot Carriers* section.

Step-by-Step Action

1) Check for legal input values. If any are out of range, output variables are set to flag values 2.0E+30 and algorithm exits.
2) Connect SMU's and Pulse Generator as in Test Configuration diagram.

   a) If pulse generator error, set values to 1.0E30, exit algorithm.
3) Force MOSFET terminal voltage.
4) Pulse gate (triangular wave with P_amp); loop for Nf # of frequencies logarithmically spaced from F_max to F_min.

   a) Measure substrate current six times and take average.

   b) If at F_max set output variable Icp.
5) Fit Qit (Qit = Icp/frequency) to the ln(frequency); calculate slope and correlation coefficient of fit.
6) Calculate Dit from slope.

   a) If Dit not in range +1E8 to +1E17, set Dit = 3.0E30.
7) Plot/save the data according to the value of Plot$.
8) Return Dit, Icp, Corcoef.

Summary of Error Codes

This value is assigned to the output variable if an error occurred during algorithm execution:

Value	Meaning
1.0E+30	Pulse generator error.
1.5E+30	Analysis not performed
2.0E+30	Input parameter error.
3.0E+30	Measurement error: Dit out of range +1E8 to +1E17
+XE+32	Operating system error "X" occurred during execution of algorithm.

Compatibility with Previous Versions

An algorithm, **CPF_MOS**, is provided to act as a parameter translation shell to maintain compatibility with HP BASIC test programs written for the obsolete **CPF_MOS**. This algorithm, which calls **CPF_3_MOS,** is briefly described next.

CPF_MOS

Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Leff	Real	µm	0.5	0~100	Effective device length
Weff	Real	µm	20	0~1E3	Effective device width
Vr†	Real	Volts	0.1	-100~100	Drain and Source bias (Vr=Vd=Vs)
Vb†	Real	Volts	0	-100~100	Bulk bias
Icp_min	Real	pA	10	1E-3~1E3	Minimum measurable charge pumping current
Vph	Real	Volts	3	-8.55~9.50[1]	Maximum pulse amplitude
Vpl	Real	Volts	-3	-8.65~9.40[1]	Minimum pulse amplitude
F_min	Real	Hz	100E3	1E3~5E6	Minimum Frequency
F_max	Real	Hz	1E6	1E3~5E6	Maximum Frequency
Nf	Integer	——	10	1~20	Number of frequencies
Integ	Integer	——	3	1~3	SMU integration level
Plot$	String	——	"Y"††		Plot/Save data

† Values are for an N-channel MOSFET. Typically, Vr for a P-channel MOSFET would be less than (or equal to) zero, Vb would be greater than (or equal to) zero.

††A "Y" plots Qit versus frequency on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files.*

1. The Range has changed for these variable to match the capability of the Agilent 8110A. It is unlikely that existing HP BASIC test programs will be affected.

Device Variables

Var Name	Var Type	Comment/Description
Width	Real	um, Mask channel width
Length	Real	um, Mask channel length
Type$	String	(N/P), Channel type

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Dit	Real	$cm^{-2}eV^{-1}$	Average interface-state density
Icp	Real	Amps	Charge pump current at F_max

Algorithm Syntax

The syntax for calling the algorithm directly is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the Introduction section.

Cpf_mos (Leff, Weff, Vr, Vb, Icp_min, Vph, Vpl, F_min, F_max, Nf, Integ,
Plot$,
Pins(*), Width, Length, Type$,
Dit, Icp)

Methodology

The algorithm is a translation shell which calls the **CPF_3_MOS** algorithm to conduct the test. The shell maintains compatibility with existing HP BASIC test programs by changing/adding the following input/output parameters in the call from **CPF_MOS** to **CPF_3_MOS**:

Input parameters:

V_ph and V_pl: Used to calculate P_amp and V_off for call to CPF_3_MOS as follows:

P_amp = ABS(Vph-Vpl)

V_off = MAX(Vph,Vpl) - P_amp/2.

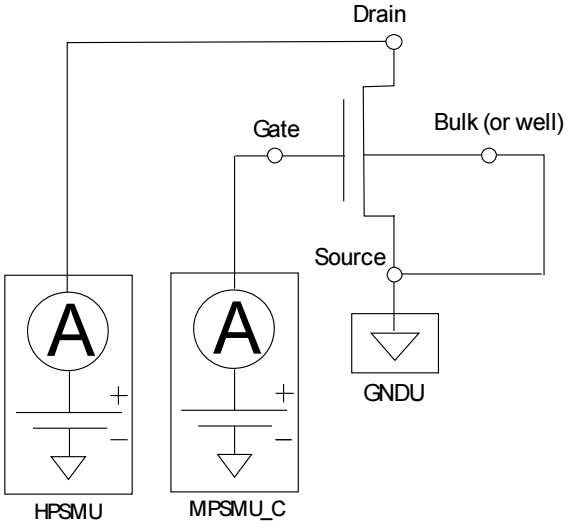
Output parameters:

Corcoef: Added but not returned to **CPF_MOS**.

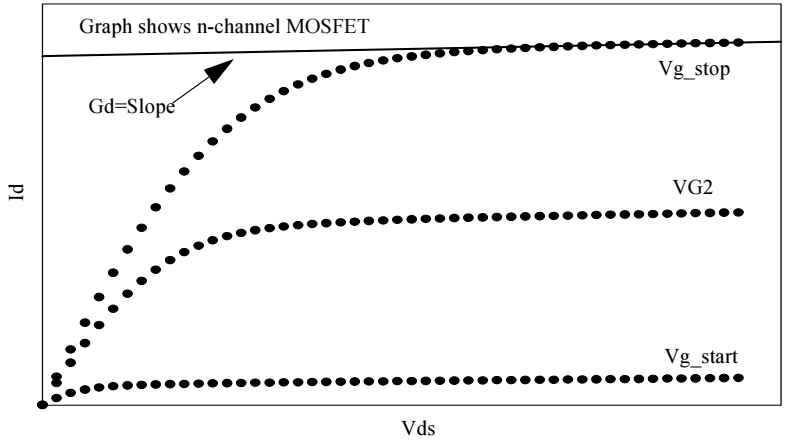
IDVD_MOS

IDVD_MOS algorithm measures the Id-Vds characteristics of a MOSFET at specified values of Vgs and determines output conductance, Gd. This variable can be an important parameter for analog applications.

Test Configuration



Electrical Characteristics



Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Vd_start†	Real	Volts	0	-100~100	Starting drain bias
Vd_stop†	Real	Volts	3.3	-100~100	Stopping drain bias
Vd_steps	Integer		50	1~500	Number of drain bias steps
Vg_start†	Real	Volts	1	-100~100	Starting gate bias
Vg_stop†	Real	Volts	3.3	-100~100	Stopping gate bias
No_vgs	Integer		2	1~20	Number of gate bias steps
Fr_meas	Integer		0	0~1	Measurement direction 0: Forward measurement 1: Reverse measurement
Nfit	Integer		8	3~100	Number of points used to obtain output conductance
Integ	Integer	——	2	1~3	SMU integration level 1: Short, 2: Medium, 3: Long
Plot$	String	——	"Y"††		Plot/save data

† Values are for an N-channel MOSFET. Typically, Vd_stop, Vg_start and Vg_stop for a P-channel MOSFET would be less than (or equal to) zero.

††A "Y" plots current versus voltage on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files*.

Device Variables

Var Name	Var Type	Comment/Description
Width	Real	um, Mask channel width
Length	Real	um, Mask channel length
Type$	String	(N/P), Channel type

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Gd	Real	µA/V	Output conductance

Algorithm Syntax

The syntax for calling the algorithm directly (that is, outside of Agilent SPECS) is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction to Hot Carriers* section.

Idvd_mos (Vd_start, Vd_stop, Vd_steps, Vg_start, Vg_stop, No_vgs, Fr_meas, Nfit, Integ, Plot$,
Pins(*), Width, Length, Type$, Gd)

Theory of Operation

The *Introduction to Hot Carriers* section provides a general overview of hot-carrier induced degradation.

IDVD_MOS measures the drain current (Ids) versus drain voltage (Vds) and returns the output conductance, Gd. The change in Gd with hot-carrier-induced damage is sometimes used for characterizing analog processes.

Methodology

IDVD_MOS measures the drain current (Ids) versus drain voltage (Vds) characteristics of a MOSFET at various gate biases (Vgs). The source and substrate of the transistor are grounded during these measurements. The drain bias is swept from Vd_start to Vd_stop in Vd_steps equally spaced steps at each gate bias and the drain current is measured at each of these drain biases. Similar measurements are performed at No_vgs equally spaced gate biases between Vg_start to Vg_stop.

Gd is calculated from the least-squares fit to the last N data points (Nfits) of the Vg_stop curve. If Vds_stop is Vdd, then Gd is the output conductance in saturation.

These measurements can be performed in the forward or reverse direction using the input parameter, Fr_meas. A forward measurement is performed with source and drain terminals of the MOSFET in the same orientation as specified in the Pins(*) array. A reverse measurement is performed with the source(drain) terminal switched with the drain(source) terminal of the device.

Forward and reverse measurements are useful in characterizing hot-carrier-induced damage. As the damage is localized near the drain-edge of the device, the comparison of forward and reverse measurements can provide insights into the nature of the damage.

Testing Notes: If **IDVD_MOS** is used as part of a hot-carrier-induced degradation experiment, and the number of Vgs steps (No_vgs) is large, this measurement can consume some time. For characterization and development experiments, it is useful to obtain this Id-Vds information over the entire operating region. For routine monitoring of Gd in hot carrier experiments, the user may wish to reduce measurement time. This can be accomplished by:

1) Reducing No_vgs to 1.
2) Increasing Vd_start to the saturation range
3) Decreasing Vd_steps.

These steps will save test time and still output Gd.

Data File Format: The input parameter Plot$ can be used to save the measured data in an ASCII file using this algorithm. The algorithm saves all the algorithm inputs and outputs into the file first. This is followed by the measured data saved in sets of two columns. Each set of data contains the Ids-Vds characteristics measured at a particular gate bias. The first column contains the drain biases while the second column contains the drain currents measured at the corresponding drain biases. A row of column headers at the top of the data identifies the parameters stored in individual columns while another line contains the gate bias at which the characteristics were performed. This format is shown in the following table:

Input parameters	
Output parameters	
$V_{gs}=V_{ds\_start}$	
V_{ds} (V)	I_{ds} (A)
<data>	<data>
$V_{gs}=V_{gs2}$	
V_{ds} (V)	I_{ds} (A)
<data>	<data>
...	

Test Structure

This routine requires and n-channel or p-channel MOSFET. See also "Test Structures" in the *Introduction to Hot Carriers* section.

Step-by-Step Action

1) Check for legal input values. If any are out of range, an error message is printed and the algorithm exits.
2) Connect SMUs and GNDU as in Test Configuration diagram.

   (a) Swap drain and source pins if a "reverse" measurement is required.
3) Set SMU integration level to Integ input value.
4) Sweep Vgs from Vg_start to Vg_stop in equal No_vgs steps.
5) At each value of Vgs, measure Ids as a function of Vds with Vds going from Vd_start to Vd_stop in Vd_steps steps.
6) Determine Gd from slope of last Nfits points of Vg_stop curve.
7) Plot/save the data according to the value of Plot$.
8) Return Gd.

Error Code Summary

This value is assigned to the output variable if an error occurred during algorithm execution:

Value	Meaning
2.0E+30	Input parameter error
1.5E+30	Analysis not performed
+XE+32	Operating system error "X" occurred during execution of algorithm

Compatibility with Previous Versions

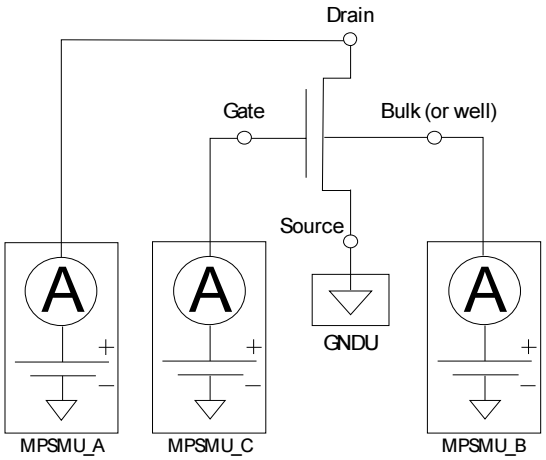
This algorithm is compatible with previous HP BASIC test programs.

S_MOS

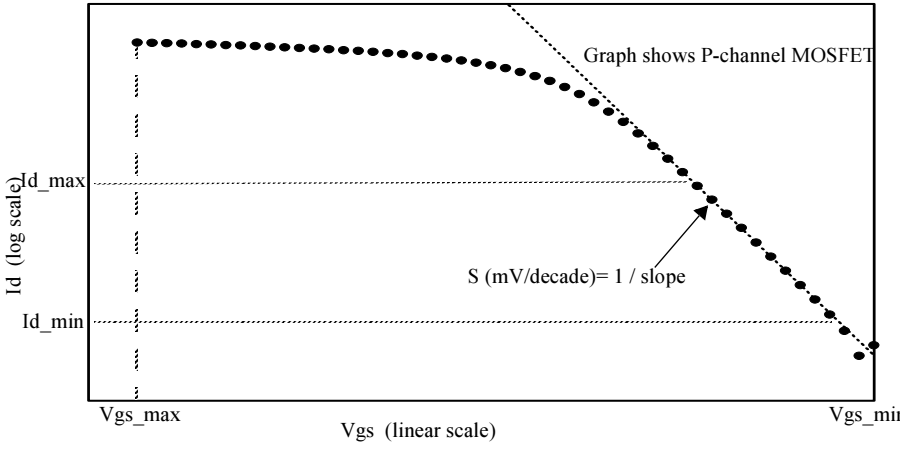
Measure subthreshold slope (S) of MOSFET device.

The subthreshold slope or "swing" is a measure of the change in gate voltage necessary to reduce drain current by one decade. Measurement should be performed in the dark to reduce light-induced source-substrate and drain-substrate diode leakage.

Test Configuration



Electrical Characteristics



Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Vds†	Real	Volts	-0.1	-100~100	Drain-to-Source bias
Vbs†	Real	Volts	0	-100~100	Bulk-to-Source bias
Vgs_min†	Real	Volts	0	-100~100	Gate-to-Source start bias
Vgs_max†	Real	Volts	-1.5	-100~100	Gate-to-Source stop bias
Id_min	Real	Amps	1E-10	1E-13~1E-3	Minimum drain current for fit
Id_max	Real	Amps	1E-7	1E-13~1E-3	Maximum drain current for fit
Fr_meas	Integer	——	0	0~1	0: Forward measurement 1: Reverse measurement
Steps	Integer	——	51	2~1001	Number of sweep steps
Integ	Integer	——	2	1~3	SMU integration level
Plot$	String	——	"Y"††		Plot/Save data

† Values are for an P-channel MOSFET. Typically, Vds for a N-channel MOSFET would be greater than zero, Vbs would be less than zero and Vgs would be greater than zero.

††A "Y" plots current versus voltage on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files*.

Device Variables

Var Name	Var Type	Comment/Description
Width	Real	um, Mask channel width
Length	Real	um, Mask channel length
Type$	String	(N/P), Channel type

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
S	Real	mV/decade	Subthreshold slope

Algorithm Syntax

The syntax for calling the algorithm directly (that is, outside of Agilent SPECS) is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction to Hot Carriers* section.

S_mos (Vds, Vbs, Vgs_min, Vgs_max, Id_min, Id_max, Fr_meas, Steps, Integ, Plot$,
Pins(*), Width, Length, Type$,
S)

Theory of Operation

The *Introduction to Hot Carriers* section provides a general overview on hot carrier degradation.

S_MOS measures a MOSFET device's subthreshold slope (S). Before threshold is reached, a MOSFET device operates like a bipolar transistor: minority carriers are injected from the source (the "emitter") into the substrate (the "base"), and these carriers are collected by the drain (the "collector"). This results in an exponential dependence of the drain current on the subthreshold gate voltage. The subthreshold slope is the inverse slope of the drain current versus the gate voltage curve:

$$S = \frac{\Delta V_{gs}}{\Delta \log_{10}(I_d)}$$

S is a fundamental device parameter which indicates how well a transistor turns off. To first order (ignoring punchthrough), S and Vtext will determine the off-state leakage current of a device. A lower value of S is better and indicates a sharper turn-off characteristic. A hot-carrier stress will change S for many technologies (due to interface-state-generation). S will also vary with the temperature of the device. For transistor device damage which is uniform along the channel (such as ionizing radiation), S can be used to extract the damage-induced interface states. For non-uniform degradation such as with channel hot carrier degradation, **CPV_3_MOS** or **CPF_3_MOS** algorithms are more effective at extracting interface states.

Methodology

S is determined by a least-squares fit to the Id-Vgs data over the user-specified range (from Id_min to Id_max).

The measurement may be made in either the forward or reverse mode. A forward measurement is made with the source and drain terminals in the same orientation as during hot-carrier stress. A reverse measurement is made with the source (and drain) terminal switched with the drain (and source) stress terminals.

Testing Notes: Measurement should be performed in the dark to reduce light induced source and drain junction leakage.

To speed measurement during process monitoring, Steps may be decreased, Vgs_max reduced and Vgs_min increased.

Data File Format: The input parameter Plot$ can be used to save the measured data in an ASCII file using this algorithm. The algorithm saves all the algorithm inputs and outputs into the file first. This is followed by the measured data saved in two columns. The first column contains the gate biases while the second column contains the drain currents measured at the corresponding gate biases. A row of column headers at the top of the data identifies the parameters stored in individual columns. This format is shown in the following table:

Input parameters	
Output parameters	
V_{GS} (V)	I_{DS} (A)
<data>	<data>

Test Structure

This routine requires an N-channel or P-channel MOSFET. See also "Test Structures" in the *Introduction to Hot Carriers* section.

Step-by-Step Action

1) Check for legal input values. If any are out of range, output variables are set to flag values 2.0E+30 and algorithm exits.
2) Connect SMUs as in Test Configuration diagram.

   a) Swap drain and source pins if a "reverse" measurement is required.
3) Set SMU integration to Integ input value.
4) Force MOSFET terminal voltages (Vbs, Vds).
5) Sweep Vgs voltage from Vgs_min to Vgs_max in increments of Steps.
6) Measure Ids at each step and build Vgs vs. Ids arrays.
7) Take absolute values of Ids array and set any zero values to resolution of the SMU.
8) If device open or short, set output to 1E+36 or 1E-99, respectively. Exit algorithm.
9) Calculate S by fit to data between Id_min and Id_max.
10) Plot/save the data according to the value of Plot$.
11) Return S.

Error Code Summary

This value is assigned to the output variable if an error occurred during algorithm execution:

Value	Meaning
1.0E-99	DUT is assumed to be shorted
1.0E+36	DUT is assumed to be open
2.0E+30	Input parameter error
1.5E+30	Analysis not performed
+XE+32	Operating system error "X" occurred during execution of algorithm

Compatibility with Previous Versions

This algorithm is compatible with previous HP BASIC test programs.

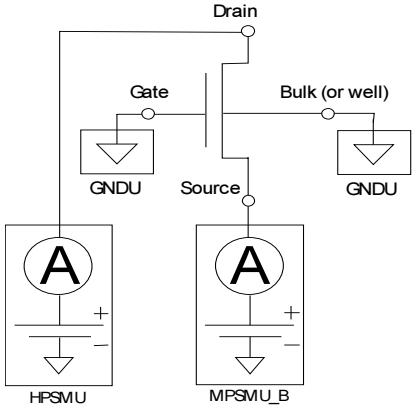
Also note that the algorithm output value of an open DUT has been changed from 1E+99 to 1E+36. This was done to maintain compatibility with Agilent SPECS maximum input value.

VPT_MOS

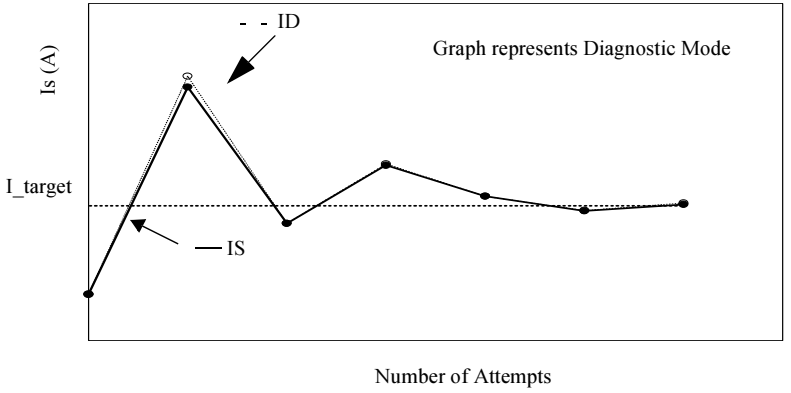
Find the voltage required for a user-specified source (or drain) current to flow when a MOSFET is "off".

VPT_MOS measures the punchthrough voltage, Vpt, of a MOSFET device. The algorithm has a diagnostic mode in which the results of a binary search for a target source current and Vpt are displayed graphically. This mode provides a method of distinguishing between punchthrough and diode breakdown. In the non-diagnostic or "stress" mode, a quasi-pulsed measurement determines Vpt based on the drain current and should be used for routine monitoring of punchthrough or during hot carrier stressing.

Test Configuration



Electrical Characteristics



Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Vds_lo[1]	Real	Volts	0	-100~100	Vd start search voltage
Vds_hi[1]	Real	Volts	-10	-100~100	Vd end search voltage
I_target[2]	Real	μA	1.0	0.01~100	Target punchthrough current
Diag	Integer	——	1	0~1	1: diagnostic mode 0: stress mode
Max_pts	Integer	——	30	10~50	Maximum points in diagnostic mode
Converge[3]	Real	%	0.5	0.1~10	Convergence window for diagnostic mode
Integ	Integer	——	2	1~3	SMU integration range 1: Short, 2: Medium, 3: Long
Plot$	String	——	"Y"		Plot/Save data

1. Values are for an P-channel MOSFET. Typically, Vds_lo and Vds_hi for an N-channel MOSFET would be greater than or equal to zero. ABS(Vds_hi) must be greater than ABS(Vds_lo); also ABS(Vds_hi) - ABS(Vds_lo) must be greater than 10 volts. Software will not necessarily apply Vds_hi. This is just the range used by the code.

2. Positive value independent of device type (N or P).

3. If this is set too low the algorithm may not converge.

††A "Y" plots current versus number of attempts on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files*.

Device Variables

Var Name	Var Type	Comment/Description
Width	Real	um, Mask channel width
Length	Real	um, Mask channel length
Type$	String	(N/P), Channel type

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Vpt	Real	Volts	Punchthrough voltage
Ipt	Real	Amps	Current at Vpt
Stat†	Real	——	Status of measurements -2 - Max_pts attempts made, no convergence -1 - user inputs below test resolution 0 - normal measurement 1 - reached compliance 2 - breakdown not reached 3 - unit oscillating 4 - measurement overflow 6 - reached time-out 7 - slew rate too small

† For additional information, see the listing of subprogram Measure_bdv in the Agilent 4062UX Programming Reference or Agilent 4070 series Programming Reference for Basic Users.

Algorithm Syntax

The syntax for calling the algorithm directly (that is, outside of Agilent SPECS) is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction to Hot Carriers* section.

Vpt_mos (Vds_lo, Vds_hi, I_target, Diag, Max_pts, Converge, Integ, Plot$,
Pins(*), Width, Length, Type$,
Vpt, Ipt, Stat)

Theory of Operation

The *Introduction to Hot Carriers* section provides a general overview of hot carrier degradation.

Algorithm **VPT_MOS** measures the punchthrough voltage (Vpt) of a MOSFET. Vpt is the drain voltage necessary for a specified current (I_target) to flow with the gate/source/substrate terminals grounded.

Vpt occurs when drain and source junction depletion regions touch. This creates a low resistance path for current to flow from the source to drain which does not depend on the gate voltage. Theoretically, if a device is at punchthrough the source and drain current will be the same. If they are not close, then another mechanism (usually avalanche breakdown of the drain to substrate diode) is occurring.

Methodology

This algorithm has two methods to determine Vpt. A "stress" mode is provided for quick measurement of Vpt. A slower "diagnostic" mode permits differentiation between punchthrough and diode breakdown.

In the "stress" mode a quasi-pulsed measurement is used to estimate Vpt by measuring the drain current. The drain voltage is ramped until the drain current reaches I_target or Vds_hi is reached. The drain voltage at this point is measured and defined as Vpt. Since only the drain current is measured, this Vpt value could be due to diode breakdown or punchthrough.

A "diagnostic" mode is provided to differentiate between punchthrough and diode breakdown. In this mode, both the drain current and source current are measured during a binary drain voltage search. Vpt is defined as the drain voltage needed to result in I_target source current. By examining the electrical characteristic the dominant breakdown mechanism may be determined. For example, if the source and drain current are not similar then diode breakdown (and not punchthrough) is occurring.

At the start of a "diagnostic" test, a "stress" quasi-pulsed measurement is performed. The Vpt value for the "stress" measurement is used to determine the upper limits of the binary voltage search. The upper limit for the binary search is the minimum of 1.1*Vpt or Vds_hi. The lower limit is Vds_lo. Typically, the first guess for Vds is (1.1*Vpt+Vds_lo)/2. This voltage is applied and the source current is measured. If this current is below I_target, the next Vds is the mid-point between the previous Vds and 1.1*Vpt. This process is repeated until the source current is within the convergence criteria of I_target (Converge) or the number of iterations exceeds Max_pts.

Since the “diagnostic” mode is relatively slow and can stress the MOSFET, it should only be used for diagnostic purposes. The “stress” mode should be used for measuring and monitoring Vpt.

Vpt decreases in P-channel devices subjected to a hot-carrier stress. The "stress" mode measurement has not been found to cause additional damage to the device. The diagnostic measurement (Diag = 1) may damage the device because of its repeated measurements. The effect of these measurements on hot-carrier-induced damage should be investigated by the user.

Testing Notes: If the Stat variable is 7, you can increase Vds_hi or increase I_target. If Stat is 1 or 3, this may indicate a bad device.

Data File Format: If the algorithm is run in diagnostic mode (Diag=1), the input parameter Plot$ can be used to save the measured data in an ASCII file using this algorithm. The algorithm saves all the algorithm inputs and outputs into the file first. This is followed by the measured data saved in two columns. The first column contains the attempts while the second column contains the drain/source currents measured for the corresponding attempt. A row of column headers at the top of the data identifies the parameters stored in individual columns. This format is shown in the following table:

Input parameters	
Output parameters	
Attempts	I_S (A)
<data>	<data>
Attempts	I_D(A)
<data>	<data>

Test Structure

This routine requires an N-channel or P-channel MOSFET of minimum channel length. Typically devices with drain channel lengths below 0.8 µm may be susceptible to low punchthrough voltage, and P-channels can have lower punchthrough voltages than N-channel devices. See also "Test Structures" in the *Introduction to Hot Carriers* section.

Step-by-Step Action

1) Check for legal input values. If any are out of range, output variables are set to flag values 2.0E+30 and algorithm exits.
2) Connect SMUs as in Test Configuration diagram.
3) Determine punchthrough voltage (Vpt):
   a) Perform a quasi-pulsed breakdown measurement (Measure_bdv) using Vds_lo, Vds_hi to determine drain voltage (Vpt) where drain current is I_target.
   b) If Diag = 1 and Stat=0, perform binary search to find drain voltage (Vpt) which results in I_target source current. Search limits are between Vds_lo and minimum of Vds_hi or 1.1*Vpt from 3a above.
   c) If Stat=0, set Vpt and Ipt to measured values. If Stat=-1 or -2, the return Vpt from last iteration. For any other Stat value set Vpt and Ipt equal to Missing.
4) If Diag=1, plot/save the data according to the value of Plot$.
5) Return Vpt, Ipt, Stat.

Error Code Summary

This value is assigned to the output variable if an error occurred during algorithm execution:

Value	Meaning
2.0E+30	Input parameter error
1.5E+30	Analysis not performed
+XE+32	Operating system error "X" occurred during execution of algorithm

Compatibility with Previous Versions

This algorithm is compatible with previous HP BASIC test programs.

Output Parameters:

Stat: Output values are expanded.

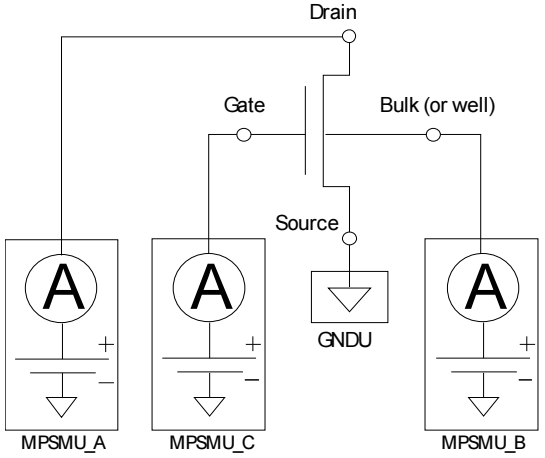
DELTAL_M2F

Extracts the difference between drawn channel length and electrical channel length, D_l, and parasitic source/drain resistance of a MOSFET device.

Using two transistors, **DELTAL_M2F** determines D_l and source/drain resistance using Whitfield's technique [1].

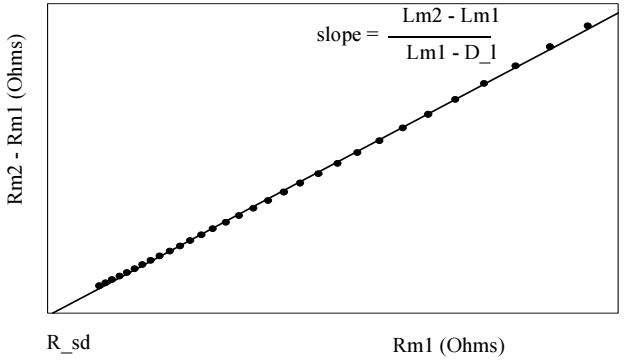
These transistors must have identical widths and the drawn channel length of the transistor 2 (Lm2) must be at least 3 times greater than Lm1 of transistor 1. To minimize error, the threshold voltage of both devices should be close. Larger values of Vgs may reduce the error due to threshold voltage mismatch. Vgs should be limited to less than 7 MV/cm to minimize gate oxide damage.

Test Configuration



transistor 1 (Lm1) & transistor 2 (Lm2)

Electrical Characteristics



Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Vgs_min†	Real	Volts	2	-100~100	Beginning gate voltage
Vgs_max†	Real	Volts	5	-100~100	Ending gate voltage
Nsteps	Integer	——	31	20~1001	Number of gate voltage steps
Vds	Real	Volts	0.025	-100~100	Linear drain voltage
Plot$	Real	——	"Y"††		Plot/Save data

† Values are for an N-channel MOSFET. Typically, Vds for a P-channel MOSFET would be less than zero, Vbs would be greater than zero and Vgs would be less than zero.

††A "Y" plots Rm2-Rm2 versus Rm1 on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files*.

Device Variables

Var Name	Var Type	Comment/Description
Width1	Real	um, Mask channel width of transistor 1
Length1	Real	um, Mask channel length of transistor 1 (smallest length)
Width2	Real	um, Mask channel width of transistor 2
Length2	Real	um, Mask channel length of transistor 2 (largest length)
Type$	String	(N/P), Channel Type

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
D_l	Real	µm	Difference in drawn and actual channel lengths
R_sd	Real	Ω	Source & Drain resistance
Corcoef	Real	——	Correlation Coefficient

Algorithm Syntax

The syntax for calling the algorithm directly (that is, outside of Agilent SPECS) is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction to Hot Carriers* section.

Deltal_m2f (Vgs_min, Vgs_max, Nsteps, Vds, Plot$,
Pins(*), Width1, Length1, Width2, Length2, Type$,
D_l, R_sd, Corcoef)

Theory of Operation

The *Introduction to Hot Carriers* section provides a general overview to transistor reliability.

DELTAL_M2F estimates the difference (D_l) between mask (Lm) and effective channel length (Leff) and parasitic source & drain resistance (R_sd) using the technique described by Whitfield [1]. D_l is a fundamental parameter for MOSFETs. Variations in D_l impact drive current (switching speed), punchthrough (leakage current) and hot-carrier-induced degradation. In a similar fashion, R_sd will impact hot-carrier-induced degradation and switching speed.

A review of techniques for estimating effective channel length is in reference [2]. Since a number of important caveats are explained, it is advised that the user review this work. Additional D_l techniques not in [2], such as the "shift and ratio" method, have also been proposed [3].

The advantage of Whitfield's test over most other D_l techniques is that it has limited assumptions, simple analysis and only requires two devices. The two assumptions in Whitfield's technique are:

1) R_sd is independent of Vgs.

2) Device threshold voltages are the same (most techniques assume this).

As a result, it is usually accurate (within its assumptions). Compared with other techniques, this approach has a simple analysis. Furthermore, it is a quicker test than techniques which test more than two devices.

[1] J. Whitfield, "A modification on 'An improved method to determine MOSFET channel length'," IEEE Trans. Electron. Dev. Letters, EDL-6, March 1985, pp. 109-110.

[2] K.K. Ng and J.R. Brews, "Measuring the effective channel length in MOSFETs," IEEE Circuits and Devices, Nov. 1990, pp. 33-38.

[3] Y. Taur, et.al., "A new shift and ratio method for MOSFET channel-length extraction," IEEE Trans. Electron. Dev. Letters, vol. 15, no. 5, May 1992, pp. 267-269.

Methodology

The output resistance (Rm=Vds/Ids) of each MOSFET is measured as a function of Vgs. The devices must be operated in linear region (Vds << Vgs - Vtext). D_l and R_sd are calculated from Rm2-Rm1 versus Rm1 curve. The range of Vgs (Vgs_min to Vgs_max) should be chosen to give a straight line (maximum correlation coefficient, Corcoef). The slope and intercept of this curve is used to calculate D_l and R_sd, respectively.

Testing Notes: A deviation in linear behavior of Rm2-Rm1 versus Rm1 curve could be caused by short channel threshold effects in the smallest channel length transistor. To increase the accuracy (straighten the curve) the user can:

1) Increase Vgs_min and Vgs_max.

2) Reduce Vds.

3) Choose a transistor 1 with Lm1 above the minimum for the technology.

Data File Format: The input parameter Plot$ can be used to save the measured data in an ASCII file using this algorithm. The algorithm saves all the algorithm inputs and outputs into the file first. This is followed by the measured data saved in two columns. The first column contains parameter Rm1 while the second column contains the value Rdiff=(Rm2-Rm1). A row of column headers at the top of the data identifies the parameters stored in individual columns. This format is shown in the following table:

Input parameters	
Output parameters	
R_{m1} (Ω)	R_{diff}=(R_{m2}-R_{m1}) (Ω)
<data>	<data>

Test Structure

The ideal test structure includes 2 similar MOS transistors which are physically close to each other. Both transistors must have the same width, W, of at least 5 microns. Transistor 2 (M2FET) has a length (Lm2) at least three times

larger than the transistor 1 length (Lm1). See also "Test Structures" in *Introduction to Hot Carriers* section.

Step-by-Step Action

1) Check for legal input values. If any are out of range, output variables are set to flag values 2.0E+30 and algorithm exits.
2) Connect SMUs as in Test Configuration diagram.
3) Sweep both transistors, one at a time, from Vgs_min to Vgs_max, calculate resistance Rm1, Rm2 and Rdiff (Rm2 - Rm1) at each Vgs.
4) Check transistors resistance and exit bad if "open" >1E9 Ohms.
5) Perform least squares fit of Rdiff vs. Rm1 data.
6) Use the slope and intercept to calculate D_l and R_sd.
7) Plot/save data according to value of Plot$.
8) Return D_l and R_sd.

Summary of Error Codes

This value is assigned to the output variable if an error occurred during algorithm execution:

Value	Meaning
2.0E+30	Input parameter error.
1.0E+31	DUT is bad (high resistance).
1.5E+30	Analysis not performed
+XE+32	Operating system error "X" occurred during execution of algorithm.

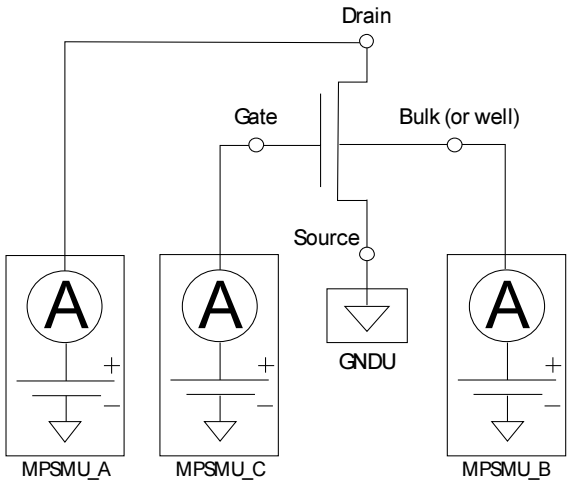
Compatibility with Previous Versions

This algorithm is compatible with previous HP BASIC test programs.

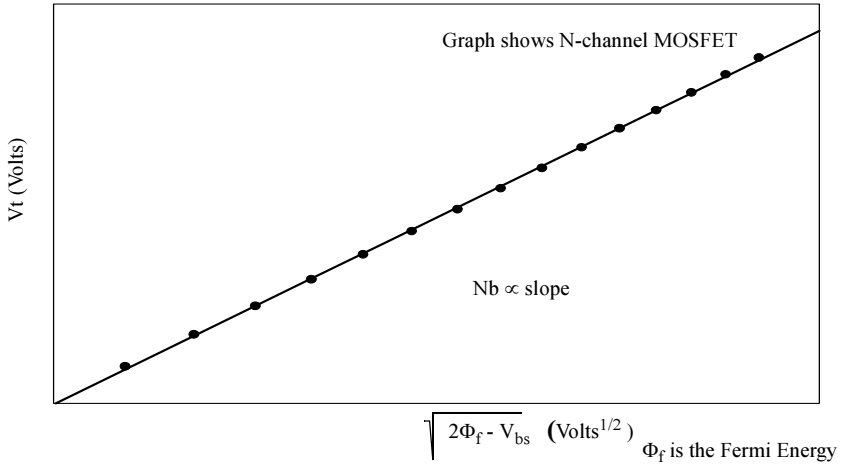
DOP_MOS

Average substrate doping of MOSFET device by measuring the change in threshold voltage with source to bulk voltage. Threshold is measured in compliance with ASTM Standard F617-86 and JEDEC Standard JESD-28 and JESD-60.

Test Configuration



Electrical Characteristics



Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Tox	Real	Å	75	10~1E3	Gate oxide thickness
Dopguess	Real	atoms/c	1E17	1E11~1E22	Estimate of doping
Vbs_min†	Real	Volts	0	-100~100	Bulk-to-Source start bias
Vbs_max†	Real	Volts	-3.5	-100~100	Bulk-to-Source stop bias
Vstep	Real	Volts	-0.25	-1~1	Step voltage for bulk bias
Vgs_min†	Real	Volts	0	-100~100	Gate-to-Source start bias
Vgs_max†	Real	Volts	3.3	-100~100	Gate-to-Source stop bias
Vds	Real	Volts	0.1	-100~100	Drain-to-Source bias for V_{text} measurement.
Steps	Integer	——	51	2~1001	Number of steps in gate sweep
Plot$	String	——	"Y"††		Plot/Save data

† Values are for an N-channel MOSFET. Typically, Vds for a P-channel MOSFET would be less than zero, Vbs_min, Vbs_max and Vstep would be greater than zero while Vgs_min and Vgs_max would be less than zero.

††A "Y" plots threshold voltage versus normalized Vbs voltage on screen. See section *PDQ-WLR Algorithm Inputs for Graphics and CSV Files*.

Device Variables

Var Name	Var Type	Comment/Description
Width	Real	um, Mask channel width
Length	Real	um, Mask channel length
Type$	String	(N/P), Channel type

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Nb	Real	atoms/cm^3	Substrate doping

Algorithm Syntax

The syntax for calling the algorithm directly (that is, outside of Agilent SPECS) is Inputs, Pins Array, Device Parameters, and Outputs. Device information for this test structure can be found in the *Introduction* section.

Dop_mos(Tox, Dopguess, Vbs_min, Vbs_max, Vstep, Vgs_min, Vgs_max,
Vds, Steps, Plot$,
Pins(*), Width, Length, Type$,
Nb)

Theory of Operation

The *Introduction to Hot Carriers* section provides a general overview of transistor and device reliability.

DOP_MOS determines the substrate doping (Nb) of a MOSFET device. The change in extrapolated threshold voltage with bulk to source voltage is used to estimate Nb. The substrate doping is a fundamental device parameter in the MOS system which impacts the threshold voltage, short channel effects, drain-induced barrier lowering and punchthrough voltage. Nb also affects the maximum lateral electric field which drives channel hot-carrier-induced damage.

The advantage of this test over **HFCV_3_CAP** in Nb determination is test speed and ability to use small in-line transistor structures.

Methodology

The general methodology for determining average substrate doping is based on the threshold voltage method outlined in [1]. The threshold voltage for a long and wide channel MOSFET (without short or narrow-width effects) may be written in the form [1]:

$$V_T = V_{FB} + 2\Phi_F + \frac{\sqrt{2\,q\varepsilon_{SI}N_b(2\Phi_F - V_{bs})}}{C_{OX}}$$

where

V_t = is the threshold voltage (in this case Vtext)

Φ_f = Fermi energy = $\frac{kT}{q}\ln\left(\frac{Nb}{ni}\right)$

V_{fb} = flatband voltage

q = electron charge

ε_{si} = permittivity of Si

V_{bs} = bulk to source voltage

C_{ox} = normalized gate oxide capacitance.

If the threshold is measured for several Vbs values, then average substrate doping, Nb, is obtained by plotting Vt versus the $\sqrt{2\Phi_f + V_{bs}}$. Since Φ_f depends on Nb and is not known in advance, a guess for Nb is used and the slope of this curve gives a new Nb value. This new Nb is used for a new Φ_f which results in an updated Nb from a fit to the curve. This process is repeated until a solution is found for Nb.

[1] D. W. Feldbaumer and D. K. Schroder, "MOSFET doping profiling," IEEE Trans. Electron Dev., vol. 38, pp. 135-140, Jan. 1991.

Testing notes: Best results are obtained when the transistor is in the linear operation region (low Vds). Depending on the range of Vbs and the doping, the Vgs_max sweep limit may be set lower than Vdd to achieve faster tests.

Data File Format: The input parameter Plot$ can be used to save the measured data in an ASCII file using this algorithm. The algorithm saves all the algorithm inputs and outputs into the file first. This is followed by the measured data saved in two columns. The first column contains the quantity $\sqrt{2\Phi_f - V_{bs}}$ for an n-channel device or $\sqrt{2\Phi_f + V_{bs}}$ for a -channel device,

while the second column contains the extrapolated threshold voltage at the corresponding Vbs. This format is shown in the following table:

Input parameters	
Output parameters	
$\sqrt{2\Phi_f - V_{bs}}$ ($V^{1/2}$)	$V_{t,ext}$ (V)
<data>	<data>

Test Structure

This algorithm requires an N-channel or P-channel MOSFET. The width and length of this device should be greater than 2 microns in order to minimize short channel and narrow width effects on the threshold voltage. Smaller channel length devices can be used but will tend to underestimate the doping. The device should have separate gate and substrate contacts although a common gate and substrate structure will usually work. See also "Test structures" in the *Introduction to Hot Carriers* section.

Step-by-Step Action

1) Check for legal input values. If any are out of range, output variables are set to flag values 2.0E+30 and algorithm exits.

2) Connect SMUs as in Test Configuration diagram.

3) Force Vds.

4) Loop through Bulk biases (Vbs stepped in Vstep increments from Vbs_min to Vbs_max).

a) Sweep gate voltage from Vgs_min to Vgs_max at each Vbs.

b) Calculate extrapolated threshold voltage (Vtext) at each Vbs.

c) If device is open or short, set output to 1.0E+31, and exit algorithm.

5) Iterate until Dopguess and resulting Nb within 0.5% or after 20 iterations

a) Use Dopguess to calculate Φ_f.

b) Fit extrapolated threshold voltage to $\sqrt{2\Phi_f - V_{bs}}$ for an N-channel MOSFET or $\sqrt{2\Phi_f + V_{bs}}$ for a P-channel MOSFET.

c) Calculate Nb from slope of least squares fit.

d) Compare Nb to Dopguess. If within 0.5%, then Nb found

e) Else use calculated Nb as new Dopguess and repeat.

6) Plot/save the data according to the value of Plot$.

7) Return Nb.

Error Code Summary

This value is assigned to the output variable if an error occurred during algorithm execution:

Value	Meaning
2.0E+30	Input parameter error
1.0E+31	DUT is bad
1.5E+30	Analysis not performed
+XE+32	Operating system error "X" occurred during execution of algorithm

Compatibility with Previous Versions

This algorithm is compatible with previous HP BASIC test programs.

HC_LINK_2_MOS

HC_LINK_2_MOS is used to provide an automatic database link between multiple stress tests and measurement algorithms. It is useful for hot carrier tests where multiple algorithms are used achieve the output. For example, this routine is used by the **HCI_3_MOS** to allow the user to quickly access lifetime data and the underlying measurements used to create it.

There is no Electrical Characteristic to plot.

Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Alg_name$	String	——	——	" "	Algorithm name or test comment
X_name$	String	——	——	" "	Independent variable name
X_value	Real	——	——	" "	Independent variable value
Units$	String	——	——		Units for X_value
Eu	Integer	——	0	0~1	X_value Flag: 1=EU (Engineering Units), 0=%
Change	Real				Lifetime Criterion
Y_name$	String	——	——	" "	Dependent variable name
Y_value	Real	——	——	" "	Dependent variable value
Vds	Real	Volts	——	——	Drain to source stress voltage
Vgs	Real	Volts	——	——	Gate to source stress voltage
Vbs	Real	Volts	——	——	Bulk to source stress voltage
Ids	Real	Amps	——	——	Absolute value drain current
Ibs	Real	Amps	——	——	Absolute value substrate current
Igs	Real	Amps	——	——	Absolute value gate current
File_ptr$†	String	——	——	" "	File pointer for lifetime relations

† This is used by HCI_3_MOS to point to the lifetime raw data and record pointers for the measurements. Leave this blank if using this routine in a Test Plan (or DieTest) for hot carrier tests.

Device Variables

Var Name	Var Type	Comment/Description
Width	Real	um, Mask channel width
Length	Real	um, Mask channel length
Type$	String	(N/P), Channel type

Output Variables

There are no output variables. If database is on (default), data are saved directly to the database in the /var/opt/WLR/database/data/measured/Hc_link_2_mos area.

Algorithm Syntax

The syntax for calling the algorithm directly (that is, outside of Agilent SPECS) is Inputs and Device Parameters.

HC_LINK_2_MOS(Alg_name$,X_name$, X_value, Units, Eu, Change, Y_name$, Y_value, Vds, Vgs, Vbs, Ids, Ibs, Igs, File_ptr$, Width, Length, Type$)

Theory of Operation

HC_LINK_2_MOS provides an automatic database link between multiple stress tests and measurement algorithms. It is useful for hot carrier, plasma damage or other tests where multiple algorithms are used to achieve the output.

Methodology

HC_LINK_2_MOS is used in **HCI_3_MOS** to quickly relate the lifetime to the underlying I-V measurements or parameters. It also allows for quick lifetime analysis. Although not recommended, this algorithm could be used in simple hot carrier tests generated by Test Plan (or DieTest). This would be done if the user wishes to store the stress voltage or other variable with the lifetime in one database record. For further information see **HCI_3_MOS**.

Test structure

There is no test structure for this algorithm.

Step-by-Step Action

1) Read input data.

2) Send to database.

HC_LINK_MOS

This routine is provided for compatibility with previous versions and is no longer a supported component. Please use HC_LINK_2_MOS.

HC_LINK_MOS is used to provide an automatic database link between multiple stress tests and measurement algorithms. It is useful for hot carrier tests where multiple algorithms are used achieve the output. For example, this routine is used by the **HCI_3_MOS** to allow the user to quickly access lifetime data and the underlying measurements used to create it.

There is no Electrical Characteristic to plot.

Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Alg_name$	String	——	——	" "	Algorithm name or test comment
X_name$	String	——	——	" "	Independent variable name
X_value	Real	——	——	" "	Independent variable value
Y_name$	String	——	——	" "	Dependent variable name
Y_value	Real	——	——	" "	Dependent variable value
Vds	Real	Volts	——	——	Drain to source stress voltage
Vgs	Real	Volts	——	——	Gate to source stress voltage
Vbs	Real	Volts	——	——	Bulk to source stress voltage
Ids	Real	Amps	——	——	Absolute value drain current
Ibs	Real	Amps	——	——	Absolute value substrate current
Igs	Real	Amps	——	——	Absolute value gate current
File_prt$†	String	——	——	" "	File pointer for lifetime relations

† This is used by HCI_3_MOS to point to the lifetime raw data and record pointers for the measurements. Leave this blank if using this routine in a Test Plan (or DieTest) for hot carrier tests.

Device Variables

Var Name	Var Type	Comment/Description
Width	Real	um, Mask channel width
Length	Real	um, Mask channel length
Type$	String	(N/P), Channel type

Output Variables

There are no output variables. If database is on (default), data is saved directly to the database in the /var/opt/WLR/database/data/measured/Hc_link_mos area.

Algorithm Syntax

The syntax for calling the algorithm directly (that is, outside of Agilent SPECS) is Inputs and Device Parameters.

HC_LINK_MOS(Alg_name$,X_name$, X_value, Y_name$, Y_value, Vds,
Vgs, Vbs, Ids, Ibs, Igs, File_ptr$,
Width, Length, Type$)

Theory of Operation

HC_LINK_MOS provides an automatic database link between multiple stress tests and measurement algorithms. It is useful for hot carrier, plasma damage or other tests where multiple algorithms are used to achieve the output.

Methodology

HC_LINK_MOS was used in PDQ-WLR Revision 2.00 **HCI_3_MOS** to quickly relate the lifetime to the underlying I-V measurements or parameters. It also allowed for quick lifetime analysis. Although not recommended, this algorithm could be used in simple hot carrier tests generated by Test Plan (or DieTest). This would be done if the user wishes to store the stress voltage or other variable with the lifetime in one database record. **HCI_3_MOS now uses HC_LINK_2_MOS to support lifetime calculations specifying the change variable in percent or engineering units**.

Test structure

There is no test structure for this algorithm.

Step-by-Step Action

1) Read input data.

2) Send to database.

PD_LINK_MOS

PD_LINK_MOS is used to provide a database table for plasma damage tests.

There is no Electrical Characteristic to plot.

Input Variables

Var Name	Var Type	Var Unit	Default Value	Range (min~max)	Comment/Description
Test_name$	String	——	——	" "	Test name
Dut_type$	String	——	——	" "	Device type, such as Ref or Poly
Ant_type	String	——	——	" "	Antenna type, such as Edge or Via
Meas_type$	String	——	——	" "	Measurement type, such as dVt
Ref_rsvp	Real	——	——	" "	Dependent variable value
Dut_rsvp	Real	——	——	" "	Dependent variable value
Vds	Real	Volts	——	——	Drain to source stress voltage
Vgs	Real	Volts	——	——	Gate to source stress voltage
Vbs	Real	Volts	——	——	Bulk to source stress voltage

Device Variables

Var Name	Var Type	Comment/Description
Width	Real	um, Mask channel width
Length	Real	um, Mask channel length
Type$	String	(N/P), Channel type

Output Variables

Var Name	Var Type	Var Unit	Comment/Description
Delta_dut_ref	Real		Dut_rsvp - Ref_rsvp

Algorithm Syntax

The syntax for calling the algorithm directly (that is, outside of Agilent SPECS) is Inputs, Device Parameters, and Outputs.

Pd_link_mos(Test_name$,Dut_type$,Ant_type$,Meas_type$, Ref_rsvp,
Dut_rsvp, Vds, Vgs,
Width, Length, Type$
Delta_dut_ref)

Theory of Operation

PD_LINK_MOS provides database records useful for plasma damage tests.

Methodology

PD_LINK_MOS is useful for plasma damage tests performed using a Test Plan (or DieTest). Appropriately applied, this would provide one record for each plasma structure (polysilicon antenna, metal antenna, etc.) and the resulting test data.

The following test shows the use of **PD_LINK_MOS** in a simple Agilent SPECS Test Plan.

Sample Plasma Damage Test in Agilent SPECS

Algorithm	Input	Output
VT_MOS_XL		A,Vtref_0,B,C
HCSTRESS_MOS_XL	Time=10	
VT_MOS_XL		D,Vtref_1,E,F
VT_MOS_XL		G,Vtant_0,H,I
HCSTRESS_MOS_XL	Time=10	
VT_MOS_XL		J,Vtant_1,K,L
PD_LINK_MOS_XL	Test_name = "Prestress", Dut_type = "Poly", Ant_type = "Edge", Meas_type = "dVt", Ref_rsvp = Vtref_0, Dut_rsvp = Vtant_0, Vds, Vgs, Vbs - same as HCSTRESS_MOS inputs	
PD_LINK_MOS_XL	Test_name = "Poststress", Dut_type = "Poly", Ant_type = "Edge", Meas_type = "dVt", Ref_rsvp = Vtref_1, Dut_rsvp = Vtant_1, Vds, Vgs, Vbs - same as HCSTRESS_MOS inputs	

Antenna Device

Reference Device

The first **PD_LINK_MOS** call at the end of the Test Plan stores the pre-stress plasma data in a single database record in the /var/opt/WLR/database/data/measured/Pd_link_mos/ database. This record consists of as-processed threshold voltages of the reference device and from the as-processed plasma device and the calculated difference between the two. The second PD_LINK_MOS call stores a second record in the database. This record consists of post-stress (hot carrier stress) data.

Test structure

There is no test structure for this algorithm.

Step-by-Step Action

1) Read input data.
2) Calculate Dut_rsvp - Ref_rsvp
2) Send to database.