In-Chassis Calibration of PMMs

16 November 2020
M. Daniels

This tutorial addresses how to automate the calibration and adjustment of the Fluke Calibration Pressure Measurement Modules ("PMM") while they remain in the chassis of a 6270A, 8270A, or 8370A. The automation scheme is similar to that of the PMM calibration when used with the Cal Sled accessory – (link: https://support.flukecal.com/hc/en-us/articles/115005079666-Macros-PM200-and-PM600-Fully-Automated-Calibration).

Like the Cal Sled concept, the In-Chassis calibration provides a fully automated process beginning with identifying the PMM to be calibrated, collecting As-Found data, making a correction based on the A/F data, and then completing the process with an As-Left verification data set. There is an optional guard banding assessment which can be used to determine final acceptance of the results. The In-Chassis calibration is applicable to all PMMs regardless of type (PM200 / PM500 / PM600) or pressure range.

The automation is attained using COMPASS for Pressure macros. They were built and tested using COMPASS for Pressure v5.0.50. An "Enhanced" license for the software is required.

Table of Contents: (Click the embedded link to jump to that section of the tutorial)

I. Macro Overview

There are four Test Macros required for the In-Chassis Calibrations. The macros can be manually added or imported by using the provided links at the end of this tutorial.

- ChassisCal_Master
- ChassisCal_PreTest
- ChassisCal_Absolute
- ChassisCal_GaugeDiff

"ChassisCal_Master": this macro is assigned as a Test Event macro as part of the COMPASS Test Definition. It is used to call the main automation macro based on the measurement mode of the PMM. It also contains subroutines used to set the calibration date and to restore the calibration should the As Left results not be desired.

"ChassisCal_PreTest": this macro is assigned as a Pre-Test macro as part of the COMPASS Test Definition. It queries the chassis for which PMMs are installed. Then it prompts the operator to select the one to be calibrated. The slot address and slot number ID are assigned and used throughout the rest of the calibration to build device-specific commands. The PMM is fixed as the active module in the chassis and identified in the data file as the one being calibrated. This enables the data directory (ie: "C:\dhi\COMPASS for Pressure\Data…") to be named by the serial number of the PPM being calibrated and not by the chassis serial number. Lastly, the fluid head is set to zero.

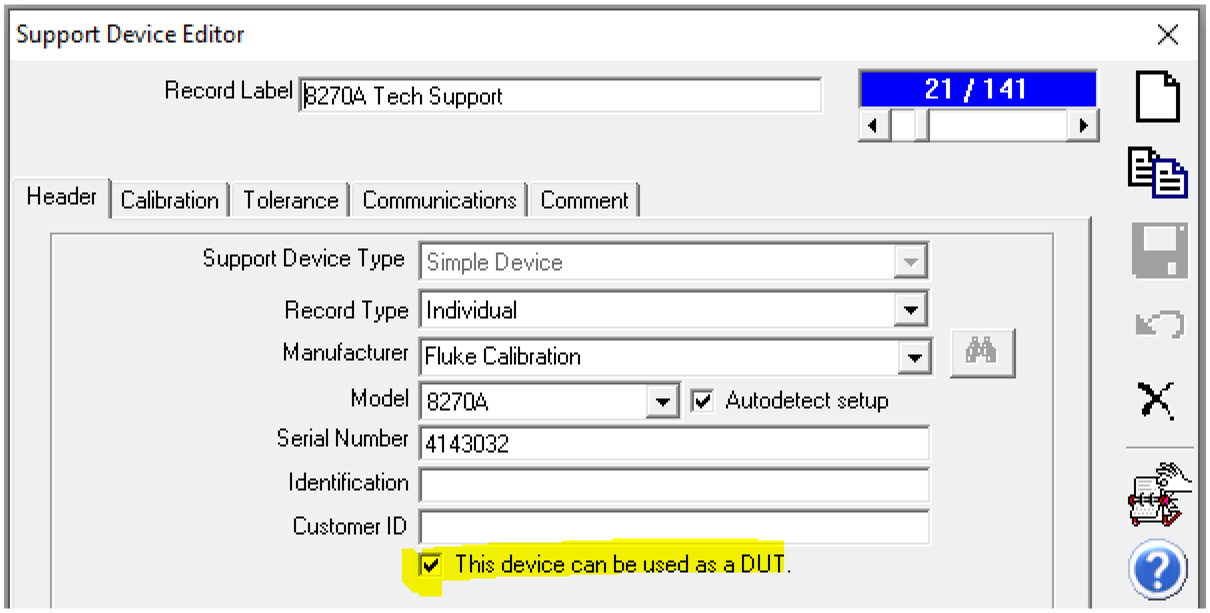
"ChassisCal_Absolute": this macro is not assigned as part of a Test Definition, but rather it is called by the ChassisCal_Master and runs in the backgroud. It is the main macro used to effect the adjustment of the PMMs operating in absolute mode. For example, calibrating a PM600-A3.5M.

"ChassisCal_GaugeDiff": this macro is not assigned as part of a Test Definition, but rather it is called by the ChassisCal_Master and runs in the backgroud. It is the main macro used to effect the adjustment of the PMMs operating in gauge or differential mode. For example, calibrating a PM200-G2M, or a PM500-BG1M.

II. COMPASS Device Definition Setups

COMPASS Device Definitions, DUT:

The DUT Definitions will follow the normal configuration methods of an Auto-Detect device. It is anticipated that most users will already have the 6270A or 8270A / 8370A configured as a Support Device. If so, then the checkbox at the bottom of the Header tab can be enabled to make the device definition useable as a DUT.

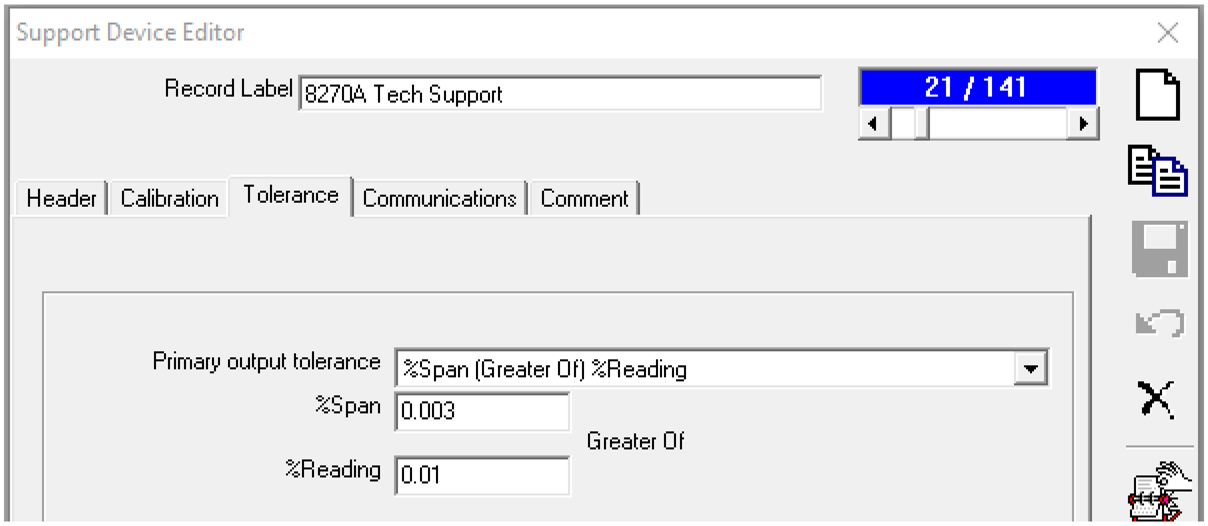

Support Device Editor

Record Label: 8270A Tech Support

21 / 141

Support Device Type: Simple Device
Record Type: Individual
Manufacturer: Fluke Calibration
Model: 8270A ☑ Autodetect setup
Serial Number: 4143032
Identification:
Customer ID:
☑ This device can be used as a DUT.

The Tolerance tab will be used by COMPASS to determine the In or Out of Tolerance status of the PMM being calibrated. The data used on the Tolerance tab needs to reflect the tolerance of the PMM being calibrated. In the situation where different types of PMMs are used in the same chassis then a decision needs to be made by each organization on how the Tolerancing will be handled. One option is to edit this information as part of the *Customize Output* during the test initialization sequence. Another option is to create individual Device Definitions for each PMM, and then have the Record Label indicate a unique identification. This tutorial will demonstrate the *Customize Output* method during test initialization. The screen capture below is for a PM600 module.


Support Device Editor

Record Label: 8270A Tech Support

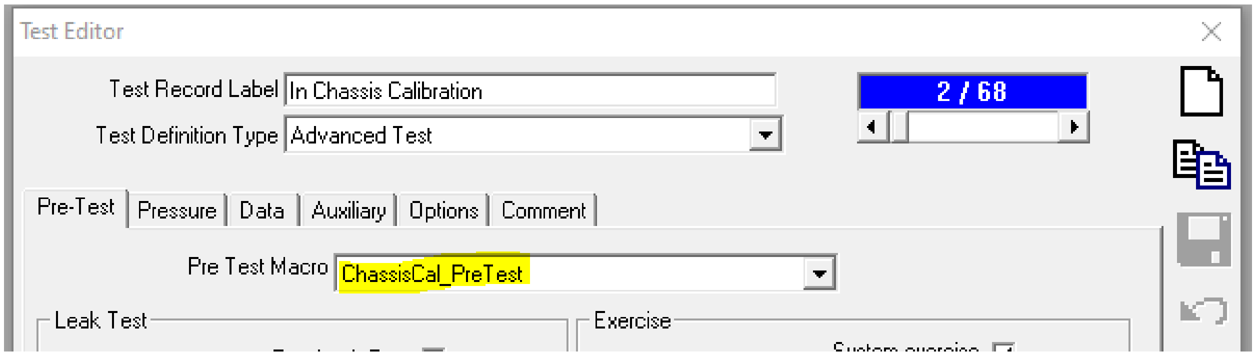
21 / 141

Primary output tolerance: %Span (Greater Of) %Reading
%Span: 0.003
Greater Of
%Reading: 0.01

<u>COMPASS Device Definitions, Support Device</u>:
The Support Device definition can be any device deemed appropriate for the calibration of a PMM. Normal methods of configuring the Support Device definition apply.

COMPASS Device Definitions, Test:

The Test Definition is the key to making the automated process work. This is where the macros are assigned and the two pressure cycles enabled. The Pre Test Macro "ChassisCal_PreTest" must be assigned on the Pre-Test tab.



Test Editor

Test Record Label: In Chassis Calibration

Test Definition Type: Advanced Test

2 / 68

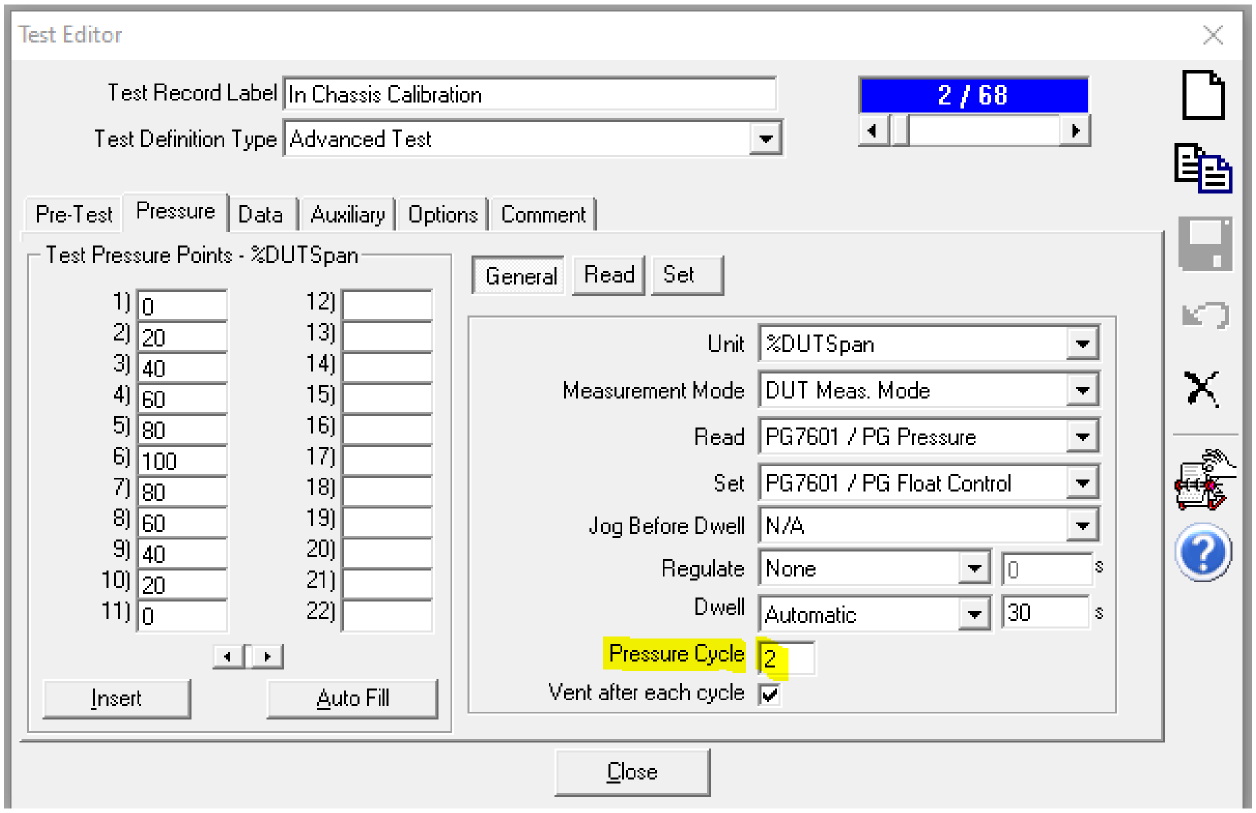
Pre-Test | Pressure | Data | Auxiliary | Options | Comment

Pre Test Macro: ChassisCal_PreTest

Leak Test

Exercise

The Pressure tab requires 2 Pressure Cyles be assigned. The first cycle is for the As Found data, and the As Left data is the second cycle. The calibration coefficients are modified between the two cycles. A typical calibration sequence may look like the following…



Test Editor

Test Record Label: In Chassis Calibration

Test Definition Type: Advanced Test

2 / 68

Pre-Test | Pressure | Data | Auxiliary | Options | Comment

Test Pressure Points - %DUTSpan

1)	0	12)	
2)	20	13)	
3)	40	14)	
4)	60	15)	
5)	80	16)	
6)	100	17)	
7)	80	18)	
8)	60	19)	
9)	40	20)	
10)	20	21)	
11)	0	22)	

Insert | Auto Fill

General | Read | Set

Unit: %DUTSpan

Measurement Mode: DUT Meas. Mode

Read: PG7601 / PG Pressure

Set: PG7601 / PG Float Control

Jog Before Dwell: N/A

Regulate: None 0 s

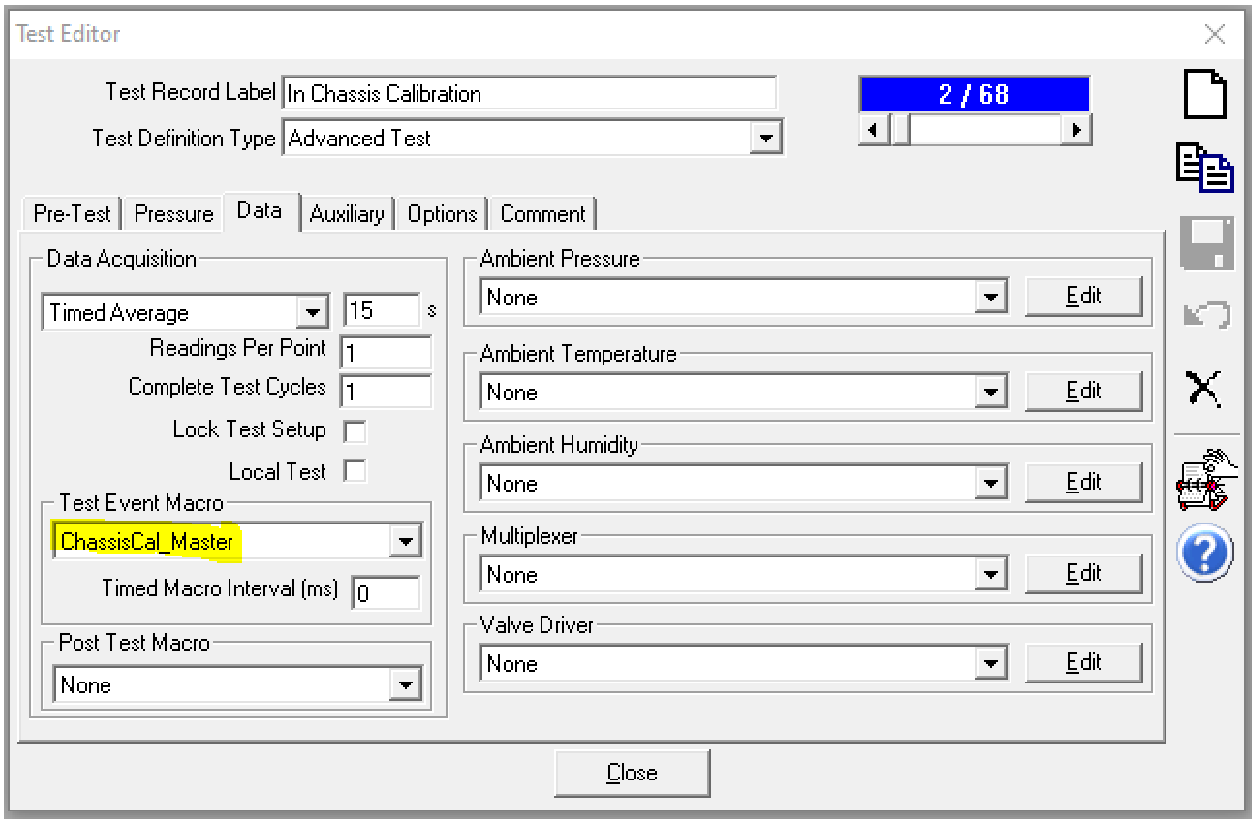
Dwell: Automatic 30 s

Pressure Cycle: 2

Vent after each cycle: ✓

Close

The Data tab is where the ChassisCal_Master macro must be assigned as a Test Event Macro:


Test Editor

Test Record Label: In Chassis Calibration

Test Definition Type: Advanced Test

2 / 68

Pre-Test | Pressure | Data | Auxiliary | Options | Comment

Data Acquisition

Timed Average 15 s

Readings Per Point: 1

Complete Test Cycles: 1

Lock Test Setup: ☐

Local Test: ☐

Test Event Macro: ChassisCal_Master

Timed Macro Interval (ms): 0

Post Test Macro: None

Ambient Pressure: None

Ambient Temperature: None

Ambient Humidity: None

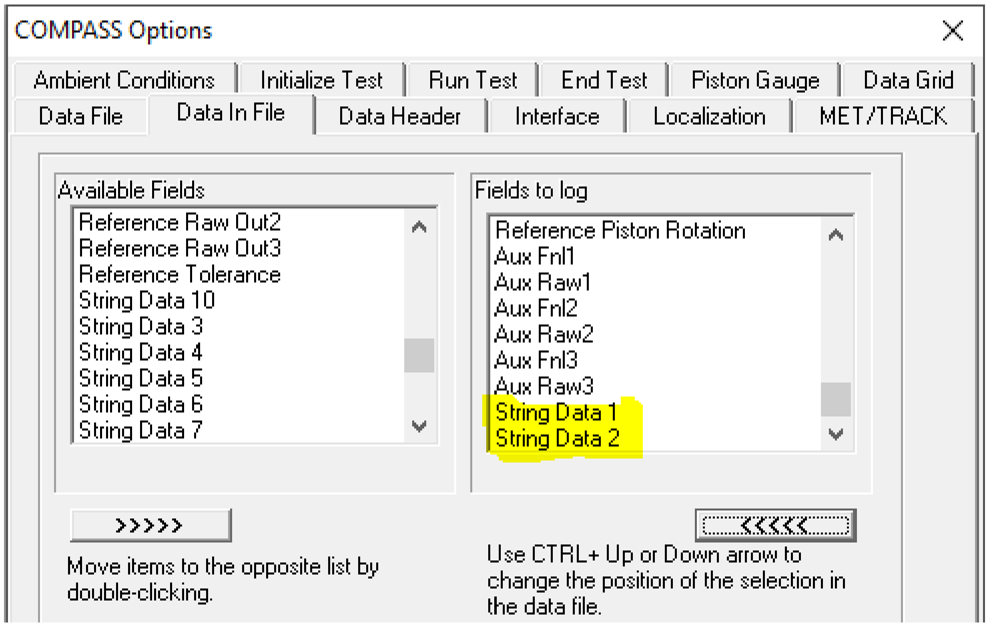
Multiplexer: None

Valve Driver: None

No Auxiliary devices are required for automated calibration.

III. COMPASS Options

There are three selections required from the [Tools],<Options> menu. In the “Data in File” tab the fields named “String Data 1”, “String Data 2” and “String Data 3” are required to be in the *Fields to log* category. These fields are used to store the PMM slot ID, Address and type.


COMPASS Options

Ambient Conditions | Initialize Test | Run Test | End Test | Piston Gauge | Data Grid

Data File | Data In File | Data Header | Interface | Localization | MET/TRACK

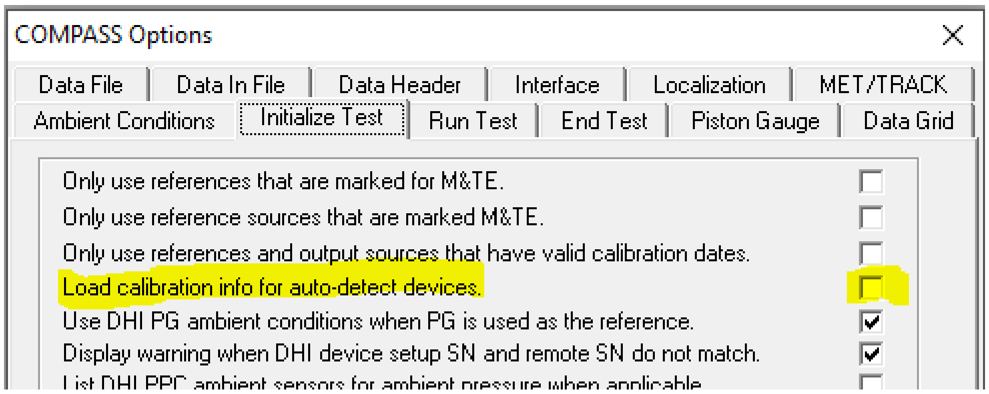
Available Fields:
Reference Raw Out2
Reference Raw Out3
Reference Tolerance
String Data 10
String Data 3
String Data 4
String Data 5
String Data 6
String Data 7

Fields to log:
Reference Piston Rotation
Aux Fnl1
Aux Raw1
Aux Fnl2
Aux Raw2
Aux Fnl3
Aux Raw3
String Data 1
String Data 2

Move items to the opposite list by double-clicking.

Use CTRL+ Up or Down arrow to change the position of the selection in the data file.

In the "Initialize Test" tab, the option to *Load calibration info for auto-detect devices* must be disabled by unchecking the box.



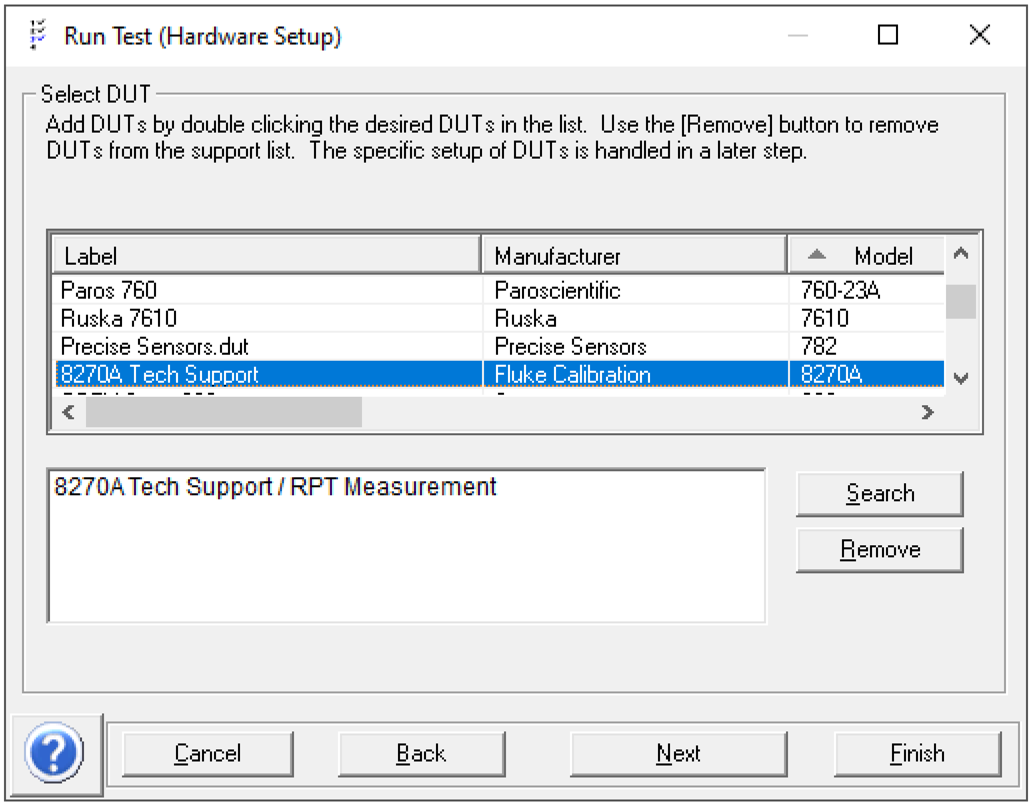
COMPASS Options

Data File | Data In File | Data Header | Interface | Localization | MET/TRACK | Ambient Conditions | Initialize Test | Run Test | End Test | Piston Gauge | Data Grid

Only use references that are marked for M&TE.	☐
Only use reference sources that are marked M&TE.	☐
Only use references and output sources that have valid calibration dates.	☐
Load calibration info for auto-detect devices.	☐
Use DHI PG ambient conditions when PG is used as the reference.	✓
Display warning when DHI device setup SN and remote SN do not match.	✓
List DHI PPC ambient sensors for ambient pressure when applicable	☐

IV. Initializing the Calibration:

In these screen shots, a PM200-G2M is being calibrated and is located in a 8270A chassis. The reference device is a PM600 and is located in a 6270A chassis. This arrangement is made for the convenience of demonstrating the process and is not an endorsement of metrological practices in terms of Test Accuracy Ratios.



Run Test (Hardware Setup)

Select DUT

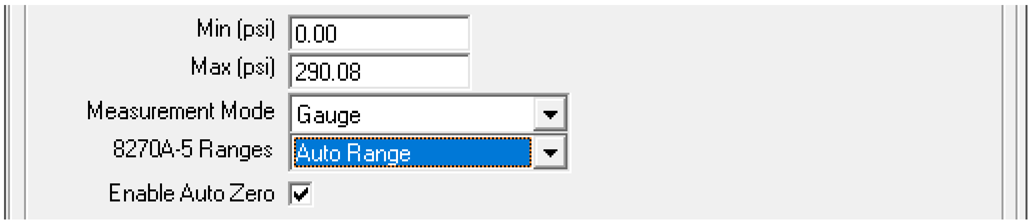
Add DUTs by double clicking the desired DUTs in the list. Use the [Remove] button to remove DUTs from the support list. The specific setup of DUTs is handled in a later step.

Label	Manufacturer	Model
Paros 760	Paroscientific	760-23A
Ruska 7610	Ruska	7610
Precise Sensors.dut	Precise Sensors	782
8270A Tech Support	Fluke Calibration	8270A

8270A Tech Support / RPT Measurement

When the instrument capabilities have been uploaded to COMPASS, then the operator is required to, 1) define the range of the PMM being calibrated, and 2) customize the Tolerance information to that of the PMM being calibrated.

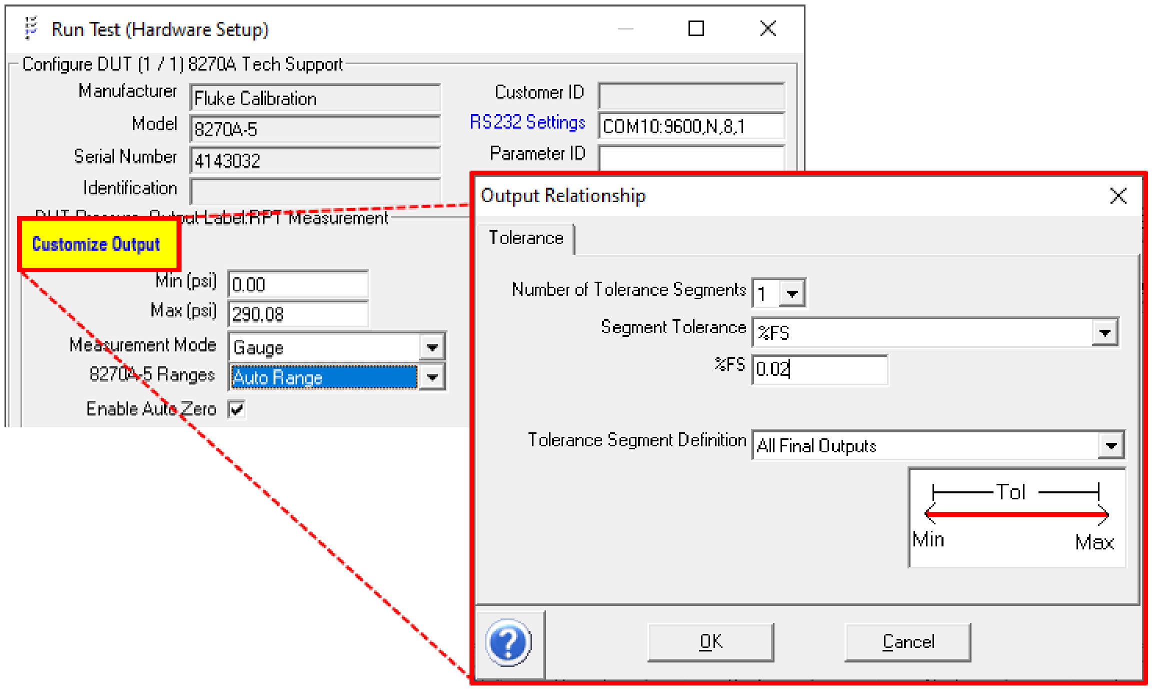
1) Define the range of the PPM to be calibrated: A key part of the In-Chassis calibration is the ability to lock in on the PMM to be calibrated. To this end, the PreTest macro will prompt the operator to (again) select the PMM to be calibrated. It's the macro-based selection which locks in the range and fixes the PMM to be used for the calibration. However, the Min and Max fields are important if the Test Definition is defined in units "% DUT Span". The option to select "Auto Range" is a valid choice.



Min (psi)	0.00
Max (psi)	290.08
Measurement Mode	Gauge
8270A-5 Ranges	Auto Range
Enable Auto Zero	✓

For gauge mode and differential mode calibrations it is important to check the box for "Enable Auto Zero". This is the mechanism which will allow the chassis to zero ("tare") the PMM when vented. For absolute mode PMMs the implications are less significant and depends on the presence and utilization of a barometric range module. The checkbox acts as an On/Off switch for the chassis to auto zero or not to auto zero. Regardless of the state of the checkbox the measured pressure will always use the PMM's zOffset coeffient.

2) Customize the Tolerance information: Although a standard feature, this method is not commonly used in COMPASS. It provides a great deal of flexibility for defining device outputs. The tolerance information is changed by clicking the blue-font text "Customize Output". This selection is important for COMPASS to determine In or Out of Tolerance, and for the correct display of the plot tolerance bars. For the PM200-G2M the tolerance is ±0.02% FS.



Configure DUT (1 / 1) 8270A Tech Support

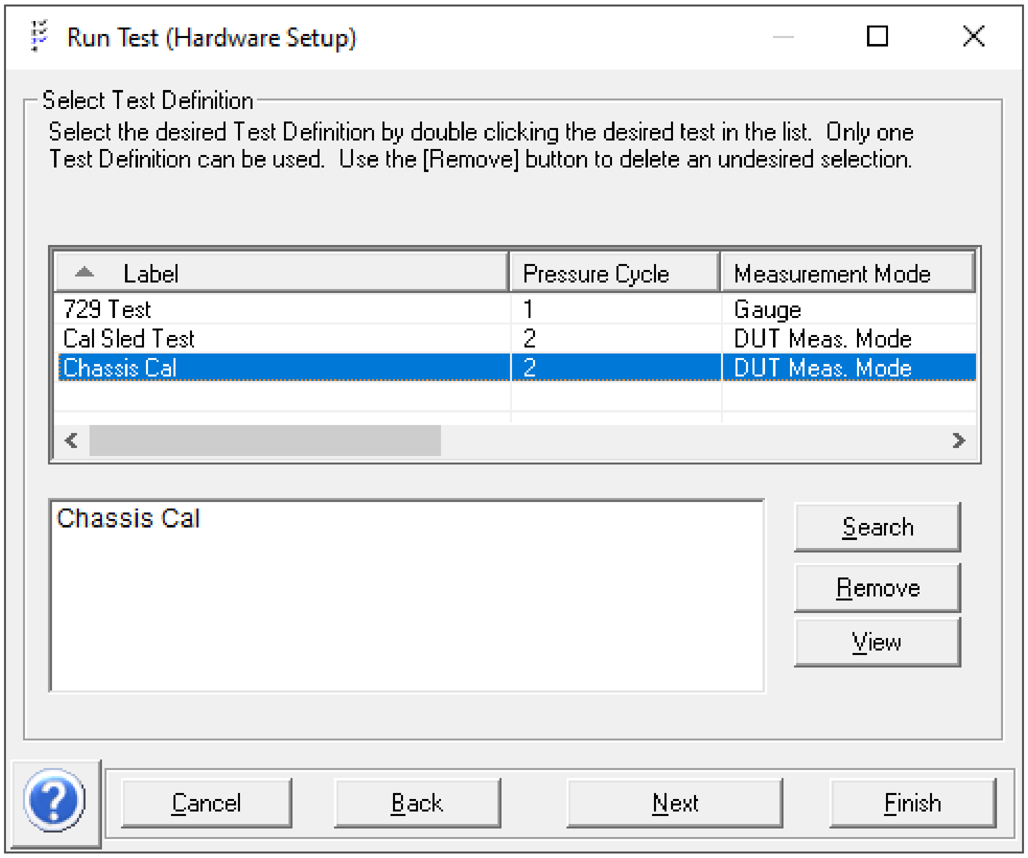
Manufacturer	Fluke Calibration	Customer ID	
Model	8270A-5	RS232 Settings	COM10:9600,N,8,1
Serial Number	4143032	Parameter ID	
Identification			

DUT Pressure Output Label:RPT Measurement

Min (psi)	0.00
Max (psi)	290.08
Measurement Mode	Gauge
8270A-5 Ranges	Auto Range
Enable Auto Zero	✓

Number of Tolerance Segments	1
Segment Tolerance	%FS
%FS	0.02
Tolerance Segment Definition	All Final Outputs

A Test Definition must be selected…



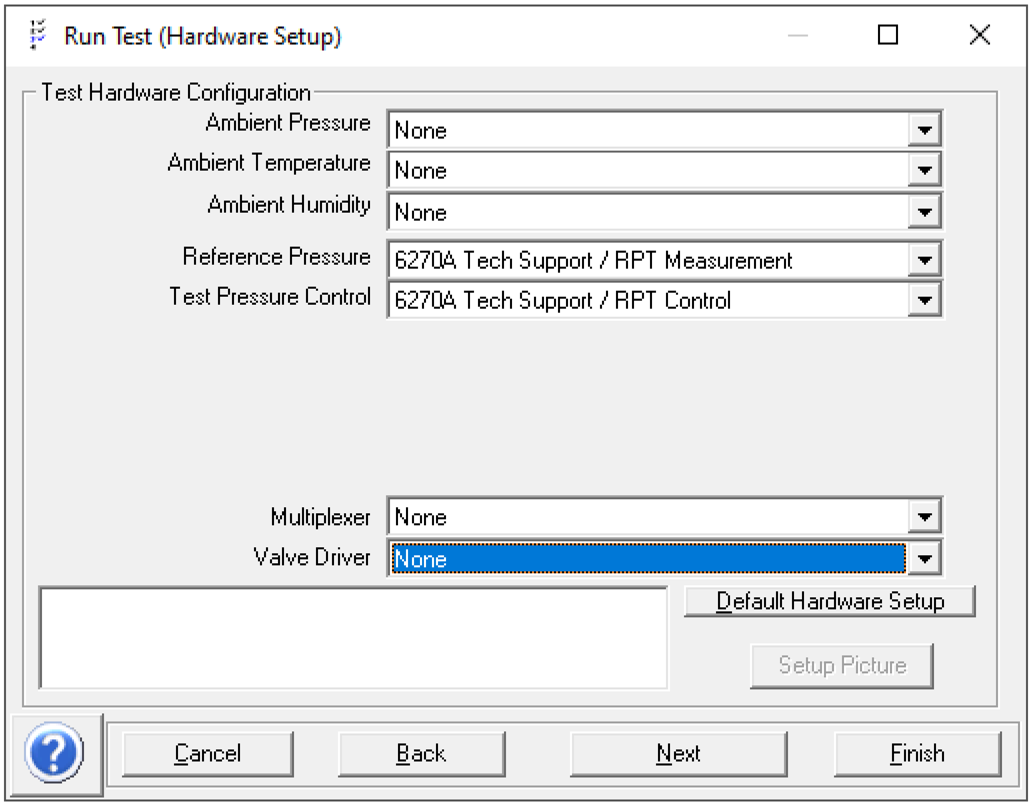
Select Test Definition

Select the desired Test Definition by double clicking the desired test in the list. Only one Test Definition can be used. Use the [Remove] button to delete an undesired selection.

Label	Pressure Cycle	Measurement Mode
729 Test	1	Gauge
Cal Sled Test	2	DUT Meas. Mode
Chassis Cal	2	DUT Meas. Mode

Chassis Cal

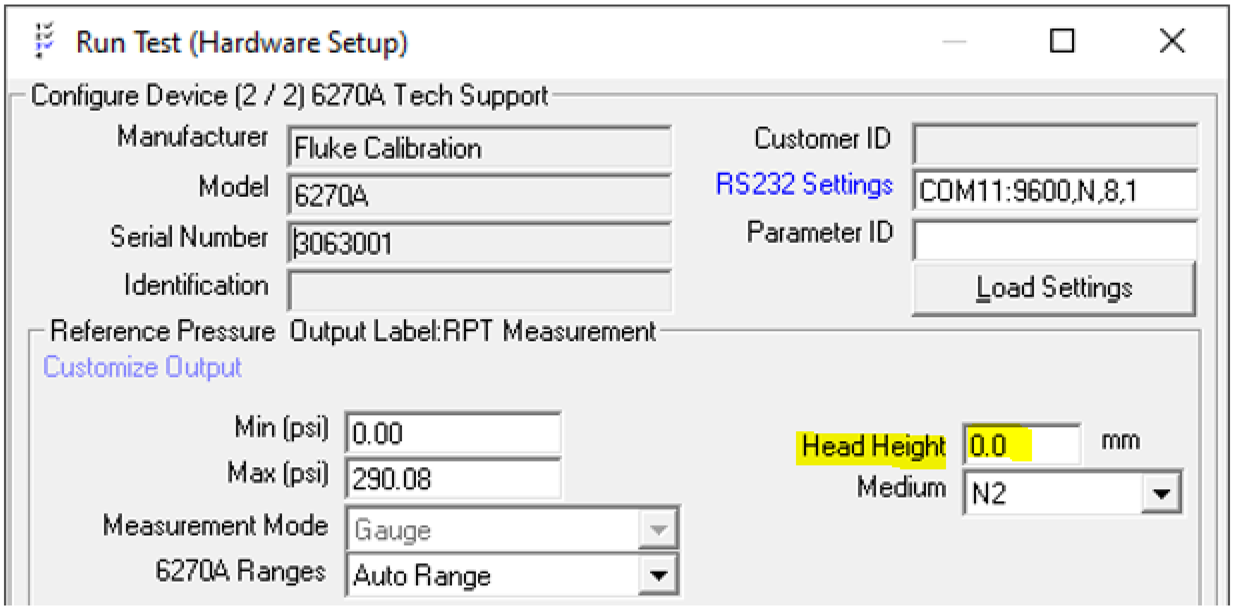
Select the Reference and Control devices…



Test Hardware Configuration

Ambient Pressure: None
Ambient Temperature: None
Ambient Humidity: None
Reference Pressure: 6270A Tech Support / RPT Measurement
Test Pressure Control: 6270A Tech Support / RPT Control
Multiplexer: None
Valve Driver: None

Set the Fluid Head height if necessary…

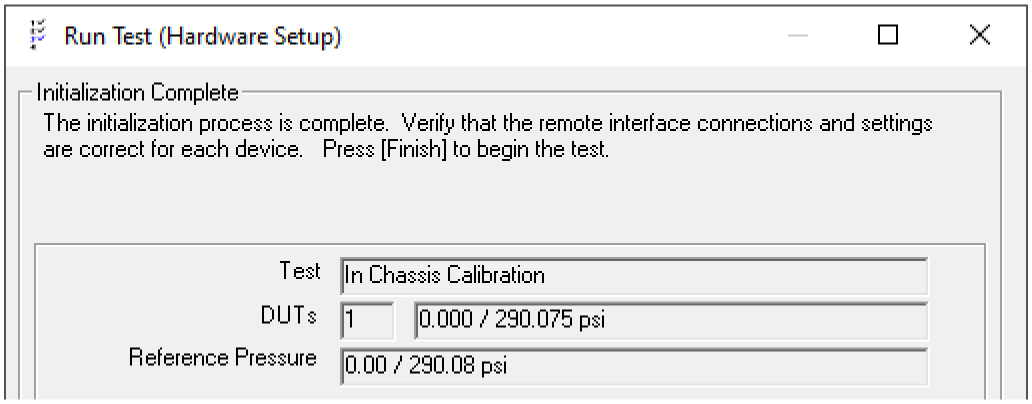

Run Test (Hardware Setup)
Configure Device (2 / 2) 6270A Tech Support
RS232 Settings
Load Settings
Customize Output

Manufacturer: Fluke Calibration
Model: 6270A
Serial Number: 3063001
Identification:
Customer ID:
RS232 Settings: COM11:9600,N,8,1
Parameter ID:

Reference Pressure Output Label:RPT Measurement

Min (psi): 0.00
Max (psi): 290.08
Measurement Mode: Gauge
6270A Ranges: Auto Range
Head Height: 0.0 mm
Medium: N2

Final confirmation of the test to be performed… Click Finish to begin the calibration.

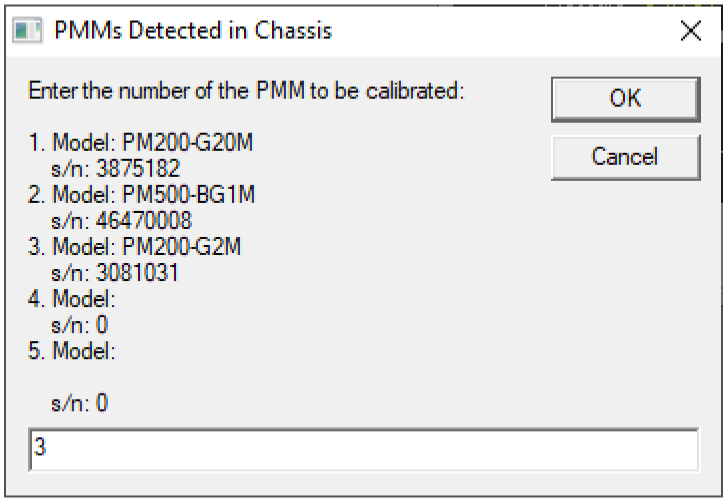


The initialization process is complete. Verify that the remote interface connections and settings are correct for each device. Press [Finish] to begin the test.

Test: In Chassis Calibration
DUTs: 1 0.000 / 290.075 psi
Reference Pressure: 0.00 / 290.08 psi

Performing the Calibration:

When the initialization steps have been completed then the automated test begins. The PreTest macro, “ChassisCal_PreTest” scans the modules installed in the chassis and prompts the operator to select the PMM to be calibrated. This user selection is used by COMPASS to make the desired PMM the “fixed” selection.



PMMs Detected in Chassis

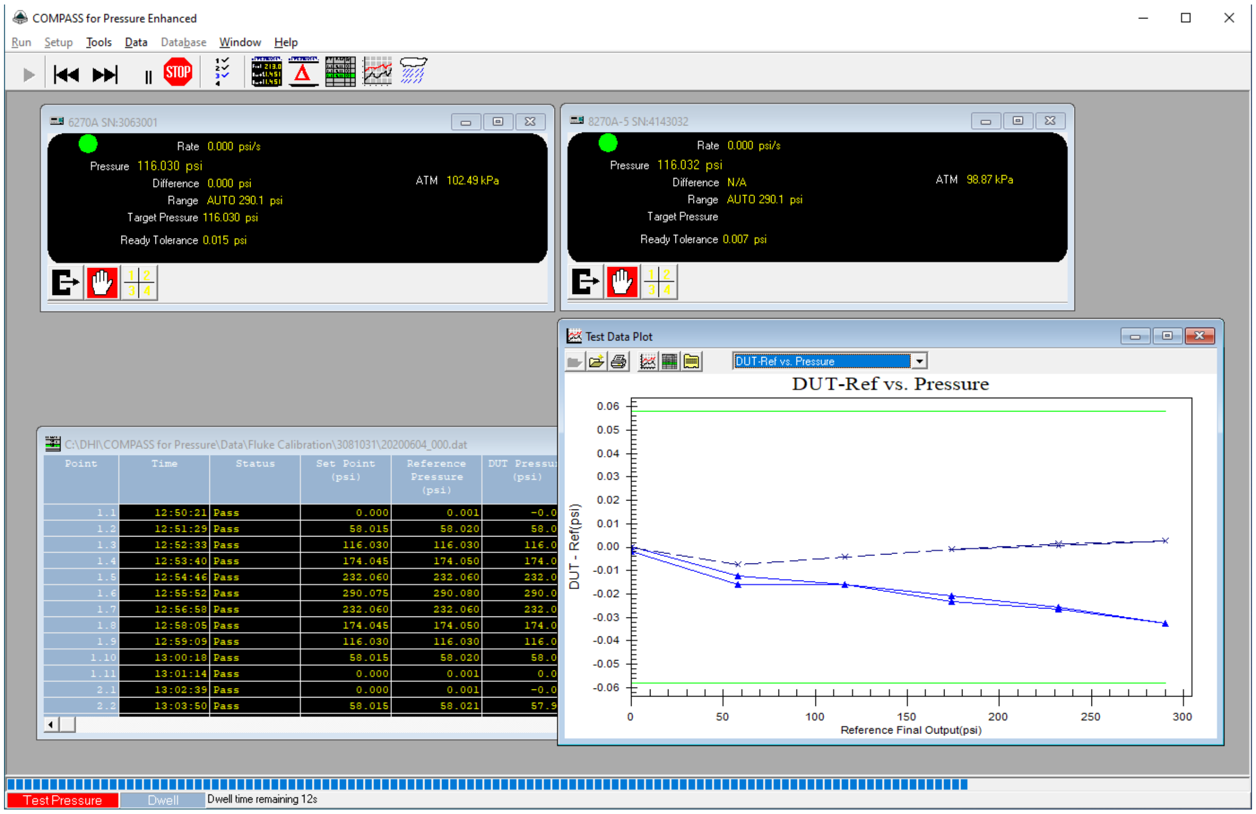
Enter the number of the PMM to be calibrated:

1. Model: PM200-G20M
   s/n: 3875182
2. Model: PM500-BG1M
   s/n: 46470008
3. Model: PM200-G2M
   s/n: 3081031
4. Model:
   s/n: 0
5. Model:

   s/n: 0

3

After the PMM selection is made then COMPASS progresses through the steps of the Test Definition. At the end of the first pressure cycle the data is automatically evaluated using a linear regression and new coefficients are determined and written to the PMM. The As Left verification cycle is then automatically started.



COMPASS for Pressure Enhanced

6270A SN:3063001

Rate 0.000 psi/s
Pressure 116.030 psi
Difference 0.000 psi
Range AUTO 290.1 psi
Target Pressure 116.030 psi
Ready Tolerance 0.015 psi
ATM 102.49 kPa

8270A-5 SN:4143032

Rate 0.000 psi/s
Pressure 116.032 psi
Difference N/A
Range AUTO 290.1 psi
Target Pressure
Ready Tolerance 0.007 psi
ATM 98.87 kPa

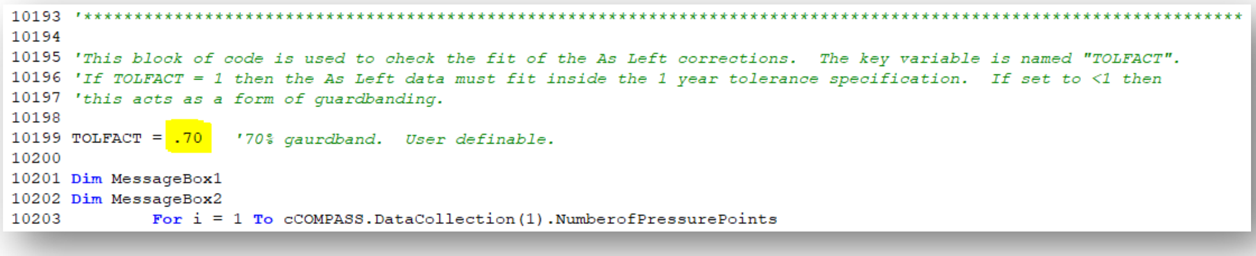
C:\DHI\COMPASS for Pressure\Data\Fluke Calibration\3081031\20200604_000.dat

Point	Time	Status	Set Point (psi)	Reference Pressure (psi)	DUT Pressu (psi)
1.1	12:50:21	Pass	0.000	0.001	-0.0
1.2	12:51:29	Pass	58.015	58.020	58.0
1.3	12:52:33	Pass	116.030	116.030	116.0
1.4	12:53:40	Pass	174.045	174.050	174.0
1.5	12:54:46	Pass	232.060	232.060	232.0
1.6	12:55:52	Pass	290.075	290.080	290.0
1.7	12:56:58	Pass	232.060	232.060	232.0
1.8	12:58:05	Pass	174.045	174.050	174.0
1.9	12:59:09	Pass	116.030	116.030	116.0
1.10	13:00:18	Pass	58.015	58.020	58.0
1.11	13:01:14	Pass	0.000	0.001	0.0
2.1	13:02:39	Pass	0.000	0.001	-0.0
2.2	13:03:50	Pass	58.015	58.021	57.9

Test Pressure Dwell Dwell time remaining 12s

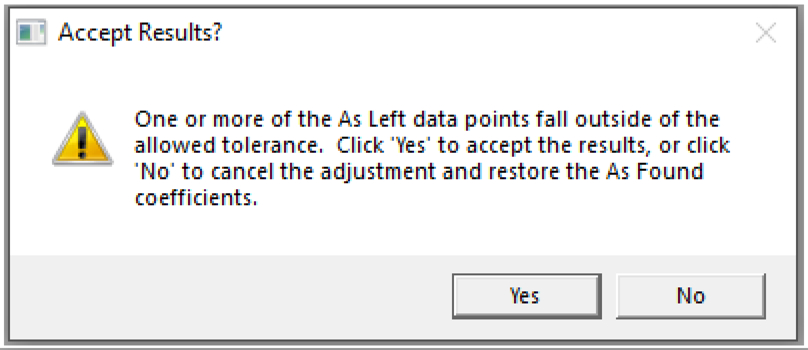
V. Restore As-Found coefficients:

The last part of the automated testing is an evaluation of the As-Left data to determine if the corrected results are inside a user-defined acceptance tolerance. The default criterion is 70% of the PMM standard specification. This is a form of guardbanding. If a different acceptance criterion is desired then the “TOLFACT” variable near the end of the ChassisCal_GaugeDiff or ChassisCal_Absolute macros can be edited. In the image below a TOLFACT of .70 is used.



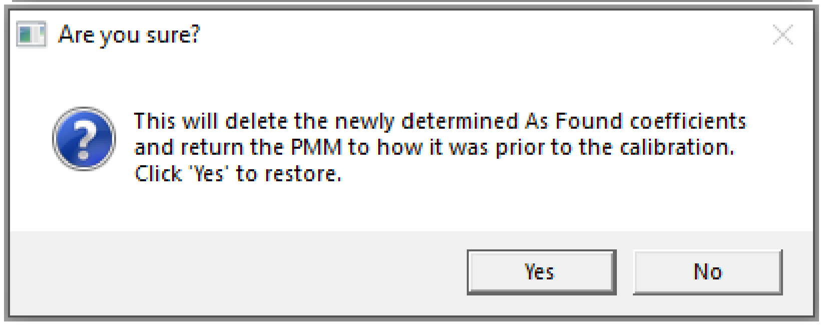
```
'*******************************************************************************************************************

'This block of code is used to check the fit of the As Left corrections.  The key variable is named "TOLFACT".
'If TOLFACT = 1 then the As Left data must fit inside the 1 year tolerance specification.  If set to <1 then
'this acts as a form of guardbanding.

TOLFACT = .70   '70% gaurdband.  User definable.

Dim MessageBox1
Dim MessageBox2
        For i = 1 To cCOMPASS.DataCollection(1).NumberofPressurePoints
```

If any part of the data set does not fall inside of the 70% guardband then a messagebox is generated alerting the user to this condition. The option is given to revert (restore) to the original As Found coefficients. The default option is to keep the newly determined coefficients. If the choice is made to restore then a second prompt is given to make sure this is the desired action. Restoring will undo the results of the test.



Accept Results?

One or more of the As Left data points fall outside of the allowed tolerance. Click 'Yes' to accept the results, or click 'No' to cancel the adjustment and restore the As Found coefficients.

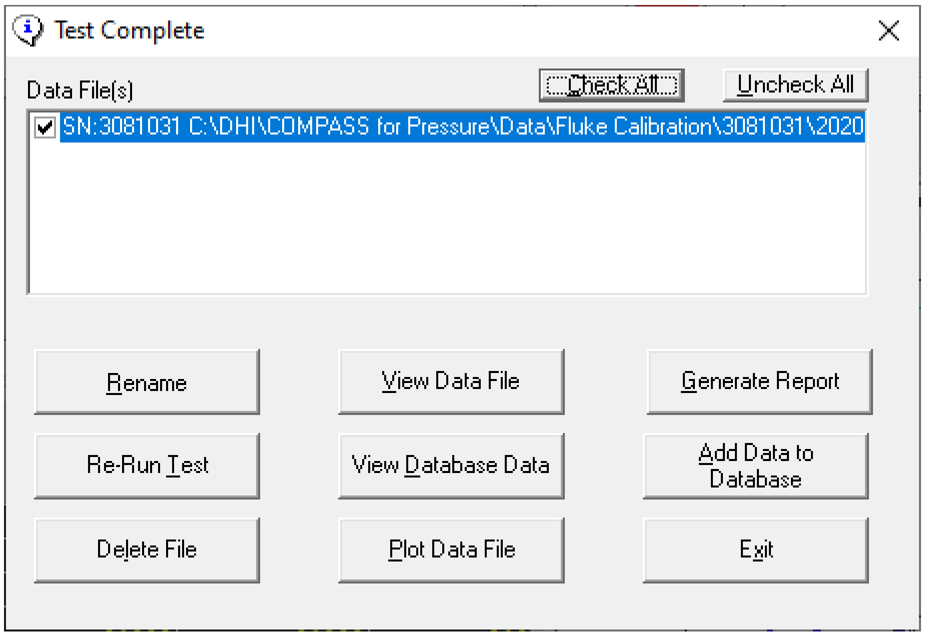


Are you sure?

This will delete the newly determined As Found coefficients and return the PMM to how it was prior to the calibration. Click 'Yes' to restore.

VI. Test Complete

When the automated testing is completed then a post test dialog box will appear. Following standard COMPASS conventions, the data file is always saved after each test point is logged.


Test Complete

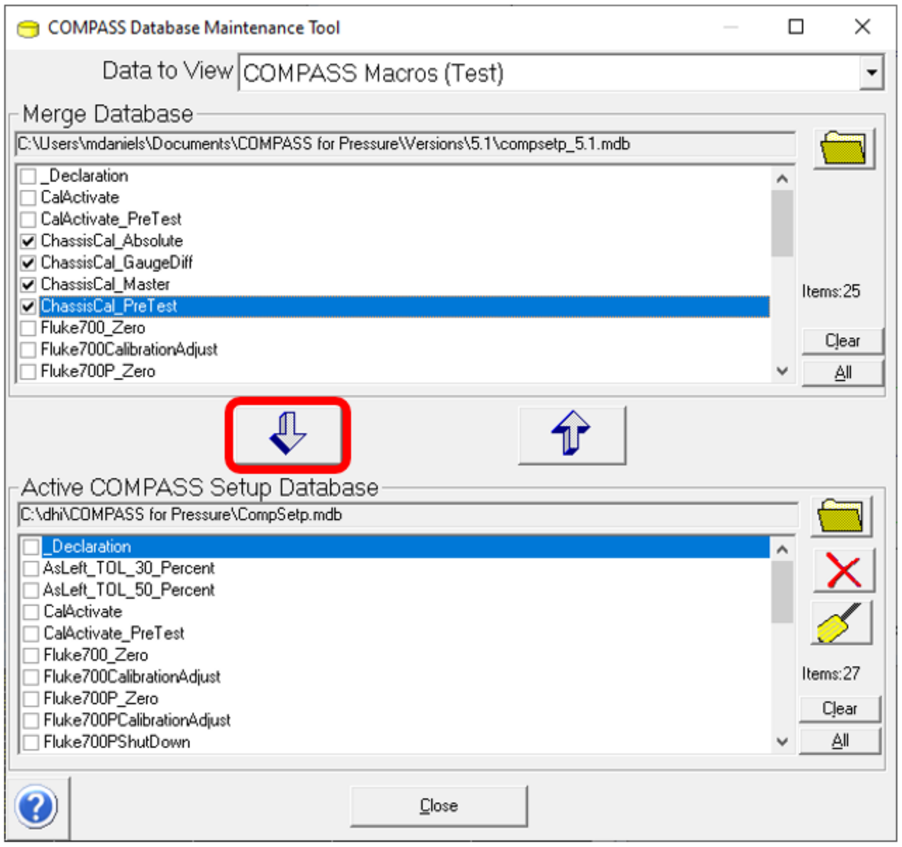
Data File(s)

☑ SN:3081031 C:\DHI\COMPASS for Pressure\Data\Fluke Calibration\3081031\2020

VII. Macro Code

There are two methods for placing the macros into [your] active database – import, or copy-and-paste.

a) Import method: the macros and example device definitions are provided as a link at the bottom of this tutorial. The link downloads a COMPASS database "compsetp_5.1.mdb". The content can be transferred in your active database using the COMPASS Database Maintenance Tool.


COMPASS Database Maintenance Tool

Data to View: COMPASS Macros (Test)

Merge Database: C:\Users\mdaniels\Documents\COMPASS for Pressure\Versions\5.1\compsetp_5.1.mdb

☐ _Declaration
☐ CalActivate
☐ CalActivate_PreTest
☑ ChassisCal_Absolute
☑ ChassisCal_GaugeDiff
☑ ChassisCal_Master
☑ ChassisCal_PreTest
☐ Fluke700_Zero
☐ Fluke700CalibrationAdjust
☐ Fluke700P_Zero

Items:25

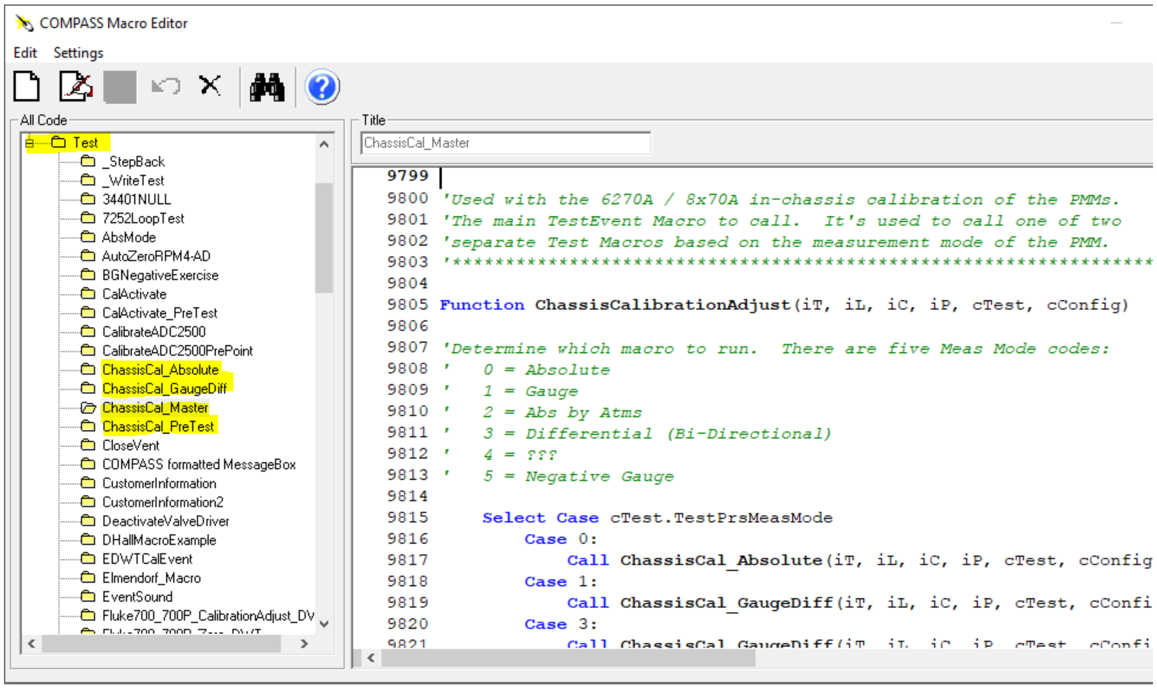
Active COMPASS Setup Database: C:\dhi\COMPASS for Pressure\CompSetp.mdb

☐ _Declaration
☐ AsLeft_TOL_30_Percent
☐ AsLeft_TOL_50_Percent
☐ CalActivate
☐ CalActivate_PreTest
☐ Fluke700_Zero
☐ Fluke700CalibrationAdjust
☐ Fluke700P_Zero
☐ Fluke700PCalibrationAdjust
☐ Fluke700PShutDown

Items:27

b) Copy-and-Paste method: VB script code is provided below which can be copied and then pasted into the COMPASS Macro Editor. New Macro folders will be required before the code can be copied.

The macros are Test macros.



```
'Used with the 6270A / 8x70A in-chassis calibration of the PMMs.
'The main TestEvent Macro to call.  It's used to call one of two
'separate Test Macros based on the measurement mode of the PMM.
'*****************************************************************

Function ChassisCalibrationAdjust(iT, iL, iC, iP, cTest, cConfig)

'Determine which macro to run.  There are five Meas Mode codes:
'   0 = Absolute
'   1 = Gauge
'   2 = Abs by Atms
'   3 = Differential (Bi-Directional)
'   4 = ???
'   5 = Negative Gauge

    Select Case cTest.TestPrsMeasMode
        Case 0:
            Call ChassisCal_Absolute(iT, iL, iC, iP, cTest, cConfig
        Case 1:
            Call ChassisCal_GaugeDiff(iT, iL, iC, iP, cTest, cConfi
        Case 3:
            Call ChassisCal_GaugeDiff(iT, iL, iC, iP, cTest, cConfi
```

ChassisCal_Master:

```
'Used with the 6270A / 8x70A in-chassis calibration of the PMMs.
'The main TestEvent Macro to call.  It's used to call one of two
'separate Test Macros based on the measurement mode of the PMM.
'**********************************************************************

Function ChassisCalibrationAdjust(iT, iL, iC, iP, cTest, cConfig)

'Determine which macro to run.  There are five Meas Mode codes:
'        0 = Absolute
'        1 = Gauge
'        2 = Abs by Atms
'        3 = Differential (Bi-Directional)
'        4 = ???
'        5 = Negative Gauge

        Select Case cTest.TestPrsMeasMode
                Case 0:
                        Call ChassisCal_Absolute(iT, iL, iC, iP, cTest, cConfig)
                Case 1:
                        Call ChassisCal_GaugeDiff(iT, iL, iC, iP, cTest, cConfig)
                Case 3:
                        Call ChassisCal_GaugeDiff(iT, iL, iC, iP, cTest, cConfig)

        End Select
'NOTE: Case 3 is the "Differential" selection in the Test Definition.  The intent
'is to start both gauge and differential mode calibrations at a vented condition
'and replicate the functionality of the user pressing the "zero" button.  Once the
'PMM is zeroed then the test moves to set the first target pressure.

End Function

Function ChassisCal_Date()
        yy = right(Year(Date),2)
        dd = Right(String(2, "0") & Day(Date), 2)
        mm = Right(String(2, "0") & Month(Date), 2)
        ChassisCal_Date = "20" & yy & "," & mm & "," & dd
End Function

'This is called by ChassisCal_Absolute and ChassisCal_GaugeDiff to restore
'the original As Found coefficients.

Function RestoreCal()
cdebug.LogStatus "The RestoreCal function has been called."

'Query the original A/F coefficients:
origC0 = cCOMPASS.DataCollection(1).DUT.CalibrationCoefficient1
origC1 = cCOMPASS.DataCollection(1).DUT.CalibrationCoefficient2
origzOffset = cCOMPASS.DataCollection(1).DUT.CalibrationCoefficient3
old_date = cCOMPASS.DataCollection(1).DUT.LastCalDate
'Convert to six fixed decimal places:
OriginalC0 = FormatNumber(origC0,6)
OriginalC1 = FormatNumber(origC1,6)
OriginalzOff = FormatNumber(origzOffset,6)
        cdebug.LogStatus "the original C0 is: " & OriginalC0
        cdebug.LogStatus "the original C1 is: " & OriginalC1
        cdebug.LogStatus "the original zOffset is: " & OriginalzOff
        cDebug.LogStatus "Calibration & Adjustment date: " & old_date

'Query the new A/L coefficients:
newC0 = cCOMPASS.DataCollection(1).DUT.CalibrationCoefficient4
newC1 = cCOMPASS.DataCollection(1).DUT.CalibrationCoefficient5
newzOffset = cCOMPASS.DataCollection(1).DUT.CalibrationCoefficient6
'Convert to six fixed decimal places:
NewC0six = FormatNumber(newC0,6)
NewC1six = FormatNumber(newC1,6)
NewzOffsix = FormatNumber(newzOffset,6)
        cdebug.LogStatus "the A/L 'NewC0six' coef is: " & NewC0six
        cdebug.LogStatus "the A/L 'NewC1six coef is: " & NewC1six
        cdebug.LogStatus "the A/L 'NewzOffsix' coef is: " & NewzOffsix

'Build query commands to help confirm the original A/F coefs have been properly written:
PMMSlotAddress = cConfig.DUTPrs(1).RangeMain.GetParent.Extra2
cmd1 = "CAL:PRES" & PMMSlotAddress & ":DATA:VAL1?"
cmd2 = "CAL:PRES" & PMMSlotAddress & ":DATA:VAL2?"
cmd3 = "CAL:PRES" & PMMSlotAddress & ":ZERO:DATA?"
cmd4 = "CAL:PRES" & PMMSlotAddress & ":DATE?"

'Build write commands to restore the original A/F coefs:
cmd5 = "CAL:PRES" & PMMSlotAddress & ":DATA:VAL1 " & OriginalC0
cmd6 = "CAL:PRES" & PMMSlotAddress & ":DATA:VAL2 " & OriginalC1
cmd7 = "CAL:PRES" & PMMSlotAddress & ":ZERO:DATA " & OriginalzOff
cmd8 = "CAL:PRES" & PMMSlotAddress & ":DATE " & old_date
```

```
        cdebug.LogStatus "cmd5 is: " & cmd5
        cdebug.LogStatus "cmd6 is: " & cmd6
        cdebug.LogStatus "cmd7 is: " & cmd7
        cdebug.LogStatus "cmd8 is: " & cmd8


Do
        'Enable the CAL Mode:
        cConfig.DUTPrs(1).IoSendCommand "CAL:MODE 1",True
        TimeDelay 1
        mode = cconfig.DUTPrs(1).IoSendCommand ("CAL:MODE?",False)
                cdebug.LogStatus "Cal Mode is: " & mode

        'Write to restore the original coefs:
        Call cConfig.DUTPrs(1).IoSendCommand(CStr(cmd5),True)
        Call cConfig.DUTPrs(1).IoSendCommand(CStr(cmd6),True)
        Call cConfig.DUTPrs(1).IoSendCommand(CStr(cmd7),True)
        Call cConfig.DUTPrs(1).IoSendCommand(CStr(cmd8),True)

        'Read back the written coefs:
        ReadBackC0raw = cConfig.DUTPrs(1).IoSendCommand (CStr(cmd1),False)
        ReadBackC1raw = cConfig.DUTPrs(1).IoSendCommand (CStr(cmd2),False)
        ReadBackZ = cConfig.DUTPrs(1).IoSendCommand (CStr(cmd3),False)
                cdebug.LogStatus "ReadBackC0raw value is: " & ReadBackC0raw
                cdebug.LogStatus "ReadackC1raw value is: " & ReadBackC1raw
                cdebug.LogStatus "ReadBackZ value is: " & ReadBackZ
        'Remove the leading coef identifier to attain only the value...
        ReadBackC0 = qextract(ReadBackC0raw,1,2,",")
        ReadBackC1 = qextract(ReadBackC1raw,1,2,",")
                cDebug.LogStatus "ReadBackC0 value is: " & ReadBackC0
                cDebug.LogStatus "ReadBackC1 value is: " & ReadBackC1

        'Check if the restored coef is different from the new A/L (it should be):
                'But first, limit each value to six decimal places...
                ReadbackC0six = FormatNumber(ReadBackC0,6)
                ReadbackC1six = FormatNumber(ReadBackC1,6)
                ReadBackZsix = FormatNumber(ReadBackZ,6)
                        cdebug.LogStatus "ReadbackC0six value is: " & ReadBackC0six
                        cdebug.LogStatus "ReadbackC1six value is: " & ReadBackC1six
                        cdebug.LogStatus "ReadBackZsix value is: " & ReadBackZsix

        C0Diff = NewC0six - ReadBackC0six
        C1Diff = NewC1six - ReadBackC1six
        ZoffDiff = NewzOffsix - ReadBackZsix
                cDebug.LogStatus "C0 restore Difference: " & C0Diff
                cDebug.LogStatus "C1 restore Difference: " & C1Diff
                cDebug.LogStatus "Zoff restore Difference: " & ZoffDiff

        If cCOMPASS.SystemAbort Then Exit Function
        If C0Diff <> 0 And C1Diff <> 0 Then Exit Do
                cdebug.LogStatus "End of Loop."

'NOTE: PM500 modules have the C0 reported in "kPa" and displayed on the chassis
'front panel in units of "psi" in the latest versions of chassis firmware.
'v1.10 = 6720A
'v1.02 = 8x70A

        cCOMPASS.StatusDisplay "Restoring the original As Found coefficients..."
Loop

End Function
```

ChassisCal_PreTest:

```
'***************************************************
'This Pre-Test macro queries the chassis for which PMMs are installed.
'Then it prompts the operator to select the one to be calibrated.  The PMM is
'fixed as the active one, identified in the .dat file as the one being calibrated,
'and the fluid head set to 0.
'***************************************************
Function GatherModuleInfo(iT, iL, iC, iP, cTest, cConfig)

'scan each slot and return the ID and serial number:
PMM1 = cConfig.DUTPrs(1).IoSendCommand ("SENS:MOD1:NAME?",False,0)
PMM1sn = cConfig.DUTPrs(1).IoSendCommand ("SENS:MOD1:SER?",False,0)
        cDebug.LogStatus "PMM1 Name: " & PMM1
        cDebug.LogStatus "PMM1 S/N: " & PMM1sn
PMM2 = cConfig.DUTPrs(1).IoSendCommand ("SENS:MOD2:NAME?",False,0)
PMM2sn = cConfig.DUTPrs(1).IoSendCommand ("SENS:MOD2:SER?",False,0)
        cDebug.LogStatus "PMM2 Name: " & PMM2
        cDebug.LogStatus "PMM2 S/N: " & PMM2sn
PMM3 = cConfig.DUTPrs(1).IoSendCommand ("SENS:MOD3:NAME?",False,0)
PMM3sn = cConfig.DUTPrs(1).IoSendCommand ("SENS:MOD3:SER?",False,0)
        cDebug.LogStatus "PMM3 Name: " & PMM3
        cDebug.LogStatus "PMM3 S/N: " & PMM3sn
PMM4 = cConfig.DUTPrs(1).IoSendCommand ("SENS:MOD4:NAME?",False,0)
PMM4sn = cConfig.DUTPrs(1).IoSendCommand ("SENS:MOD4:SER?",False,0)
        cDebug.LogStatus "PMM4 Name: " & PMM4
        cDebug.LogStatus "PMM4 S/N: " & PMM4sn
PMM5 = cConfig.DUTPrs(1).IoSendCommand ("SENS:MOD5:NAME?",False,0)
PMM5sn = cConfig.DUTPrs(1).IoSendCommand ("SENS:MOD5:SER?",False,0)
        cDebug.LogStatus "PMM5 Name: " & PMM5
        cDebug.LogStatus "PMM5 S/N: " & PMM5sn


'Present Inputbox message listing available PMMs, asking the user to make a selection:
With cConfig.DUTPrs(1).RangeMain.GetParent
        msg = "Enter the number of the PMM to be calibrated: " & vbcrlf
        msg = msg & vbcrlf
        msg = msg & "1. Model: " & PMM1 & vbcrlf
        msg = msg & "    s/n: " & PMM1sn & vbcrlf
        msg = msg & "2. Model: " & PMM2 & vbcrlf
        msg = msg & "    s/n: " & PMM2sn & vbcrlf
        msg = msg & "3. Model: " & PMM3 & vbcrlf
        msg = msg & "    s/n: " & PMM3sn & vbcrlf
        msg = msg & "4. Model: " & PMM4 & vbcrlf
        msg = msg & "    s/n: " & PMM4sn & vbcrlf
        msg = msg & "5. Model: " & PMM5 & vbcrlf
        msg = msg & "    s/n: " & PMM5sn

        reply = inputbox(msg,"PMMs Detected in Chassis", "")
        cDebug.LogStatus "The selection entered: " & reply

        'make sure a PMM was selected by the user:
        If reply = "" Then
                cdebug.LogStatus "A user input for the PMM selection was not detected. Aborted the test."
                msg = msgbox("A user input for the PMM selection was not detected. Aborting the test." ,
0+16+4096, "Test Aborted")
                cCOMPASS.SystemAbort = True
        End If


End With

'Based on the user input, select the PMM and fix it as active PMM:
If reply = 1 Then
        PMMSlotAddress = 5
        FixPMM = cConfig.DUTPrs(1).IoSendCommand ("SENS:MOD 1",True,0)
        cCOMPASS.TimeDelay 2
        ElseIf reply = 2 Then
        PMMSlotAddress = 15
        FixPMM = cConfig.DUTPrs(1).IoSendCommand ("SENS:MOD 2",True,0)
        cCOMPASS.TimeDelay 2
        ElseIf reply = 3 Then
        PMMSlotAddress = 25
        FixPMM = cConfig.DUTPrs(1).IoSendCommand ("SENS:MOD 3",True,0)
        cCOMPASS.TimeDelay 2
        ElseIf reply = 4 Then
        PMMSlotAddress = 35
        FixPMM = cConfig.DUTPrs(1).IoSendCommand ("SENS:MOD 4",True,0)
        cCOMPASS.TimeDelay 2
        ElseIf reply = 5 Then
        PMMSlotAddress = 45
        FixPMM = cConfig.DUTPrs(1).IoSendCommand ("SENS:MOD 5",True,0)
        cCOMPASS.TimeDelay 2
End If
```

```
PMMSlotNum = cConfig.DUTPrs(1).IoSendCommand ("SENS:MOD?",False,0)
cDebug.LogStatus "The fixed PMM Slot is: " & PMMSlotNum
cDebug.LogStatus "The address of the PMM to be calibrated: " & PMMSlotAddress

'SlotNumber = 1-5.  SlotAddress = 5-45.
'write the slot and address of the PMM to the .dat file:
cConfig.DUTPrs(1).RangeMain.GetParent.Extra1 = PMMSlotNum
cConfig.DUTPrs(1).RangeMain.GetParent.Extra2 = PMMSlotAddress

'Set fluid head to 0. (COMPASS should have already done this as part of DUT initialization):
cConfig.DUTPrs(1).IoSendCommand "SENS:REF:HEIG 0",True,0

'Save the the .dat file which PMM was selected:
        'Re-query the info:
        Model = cConfig.DUTPrs(1).IoSendCommand ("SENS:MOD" & reply & ":NAME?",False)
        SN = cconfig.DUTPrs(1).IoSendCommand ("SENS:MOD" & reply & ":SER?",False)
                cdebug.LogStatus "the Model selected: " & Model
                cdebug.LogStatus "the sn selected: " & SN

        'write the info to the .dat file:  (Note - this will be used by Windows to create / select
        'a new sub directory based on the PMM's serial number)
        cConfig.DUTPrs(1).RangeMain.GetParent.Model = Model
        cConfig.DUTPrs(1).RangeMain.GetParent.SN = SN

'Define a Model Type variable for use in the unit conversion needs:
ModelType = Left(Model,5)
cdebug.LogStatus "the Model type selected is:" & ModelType

If ModelType = "PM200" Then
        ModelTypeID = 2
        cConfig.DUTPrs(1).RangeMain.GetParent.Extra3 = ModelTypeID
        ElseIf ModelType = "PM500" Then
        ModelTypeID = 5
        cConfig.DUTPrs(1).RangeMain.GetParent.Extra3 = ModelTypeID
        ElseIf ModelType = "PM600" Then
        ModelTypeID = 6
        cConfig.DUTPrs(1).RangeMain.GetParent.Extra3 = ModelTypeID
End If
cdebug.LogStatus "The ModelTypeID is: " & ModelTypeID
End Function
```

ChassisCal_GaugeDiff:

```
'*******************************************************************************
'Used with ChassisCal_Master to calibrate and adjust Gauge and Differential mode PMMs.
'Applicable for 6270A / 8270A / 8370A.  Called by the master test macro "ChassisCal_Master".

'Gauge Mode testing must be performed with the user-option "Load calibration info for auto-detect
'devices" disabled.

'*******************************************************************************

Function ChassisCal_GaugeDiff(iT, iL, iC, iP, cTest, cConfig)
Dim DUT_Raw_Data()
Dim Reference_Data()
Dim C1_Corrected_Data()
Dim res(4,7)

C0 = 0
C1 = 0
zOffset = 0


'<currentteststep 1010 - record AF coefs.>
If cCOMPASS.CurrentTestStep = 1010 Then ' Test Definition data files were created for each DUT.
    cDebug.LogStatus "Starting <step 1010>  "
    cDebug.LogStatus "Capture As Found Coeffs..."

         'Ensure the Chassis is in Gauge mode:
         cConfig.DUTPrs(1).IoSendCommand "SENS:MODE GAUGE",True

         PMMSlotAddress = cConfig.DUTPrs(1).RangeMain.GetParent.Extra2
                  cDebug.LogStatus "The PMM slot address is: " & PMMSlotAddress

         cmd1 = "CAL:PRES" & PMMSlotAddress & ":DATA:VAL1?"
         cmd2 = "CAL:PRES" & PMMSlotAddress & ":DATA:VAL2?"
         cmd3 = "CAL:PRES" & PMMSlotAddress & ":ZERO:DATA?"
         cmd4 = "CAL:PRES" & PMMSlotAddress & ":DATE?"

'Identify the unit of measure being used for the test and the PMM Type:
DUTunit = cCOnfig.DUTPrs(1).RangeMain.UnitFinal
DUTunitText = cCOnfig.DUTPrs(1).RangeMain.UnitFinalText
ModelTypeID = cConfig.DUTPrs(1).RangeMain.GetParent.Extra3

         ' Read A/F C0 from PMM; make unit conversions, write to calcoef1:
         C0raw = cConfig.DUTPrs(1).IoSendCommand(CStr(cmd1), False)
                  cDebug.LogStatus "the raw C0 string is: " & C0raw
         C0 = qextract(C0raw,1,2,",")
                  If ModelTypeID = 5 Then  'this is a PM500, therefore the C0 is in units of kPa
                           cdebug.LogStatus "the A/F C0 value in kPa is: " & C0
                  C0cnvt = cCOMPASS.UnitConversion(CDbl(C0),CInt(DUTunit),3,0)
                           cdebug.LogStatus "the A/F C0 value converted to the DUT unit of measure is: " &
C0cnvt & DUTunitText
                  Else
                  'PM200 and PM600 always read the C0 in units of psi
                  cDebug.LogStatus "The C0 value before conversion is: " & C0
                  C0cnvt = cCOMPASS.UnitConversion(CDbl(C0),CInt(DUTunit),9,0)
                           cdebug.LogStatus "the A/F C0 value converted to the DUT unit of measure is: " &
C0cnvt & DUTunitText
                  End If
         cCOMPASS.DataCollection(1).DUT.CalibrationCoefficient1 = C0cnvt

         ' Read A/F C1 from DUT; write to calcoef2
         C1raw = cConfig.DUTPrs(1).IoSendCommand(CStr(cmd2), False)
         C1 = qextract(C1raw,1,2,",")
         cCOMPASS.DataCollection(1).DUT.CalibrationCoefficient2 = C1
                  cDebug.LogStatus "As Found C1 coefficient: " & C1

         ' Read A/F zOffset from PMM, write to calcoef3:
         zOffset = cConfig.DUTPrs(1).IoSendCommand(CStr(cmd3), False)
         cCOMPASS.DataCollection(1).DUT.CalibrationCoefficient3 = zOffset
                  cDebug.LogStatus "As Found zOffset is: " & zOffset
                  cDebug.LogStatus "the zOffset value is in units of: " &
cCOnfig.DUTPrs(1).RangeMain.UnitFinalText

         ' Read the last saved cal edit date:
         calDate = cConfig.DUTPrs(1).IoSendCommand(CStr(cmd4), False)
         cCOMPASS.DataCollection(1).DUT.LastCalDate = calDate
                  cDebug.LogStatus "As Found last cal date: " & calDate


         ' Place the 6270A into MEASURE mode to prepare for applying the pressure:
         cConfig.DUTPrs(1).IoSendCommand "OUTP:MODE MEASURE",True,0
         TimeDelay 2
         mode = cCOnfig.DUTPrs(1).IoSendCommand("OUTP:MODE?",False,0)
```

```
                  cdebug.LogStatus "Measurement mode: " & mode
                  cdebug.LogStatus "Placed the 6270A into MEASURE mode"

         cdebug.LogStatus "<End of CurrentTestStep 1010>"

'<The test definition will now run the first cycle of points>
'<Continue with pressure cycle 1 (as-found data)>


'Vent the 6270A and allow it to run AutoZero...
ElseIf cCOMPASS.CurrentTestStep = 1100  Then
         cDebug.LogStatus "starting step <1100>..."
         DAQ = cCOMPASS.cConfig.SetPrs(1).RangeMain.GetParent.DaqType

             If DAQ <> 0 Then
                           'Vent both the Reference and the chassis (DUT). The chassis automatically zeros the
PMM when vented:
                           cCOMPASS.cConfig.SetPrs(1).IoSetOutput 0,0,1
                           cCOMPASS.cConfig.DUTPrs(1).IoSetOutput 0,0,1
                                    cDebug.LogStatus "Sent ioSetOutput 0,0,1 to vent the Ref Device."
                                    cDebug.LogStatus "Sent ioSetOutput 0,0,1 to vent the chassis (DUT)."
                           TimeDelay 3
                           cCOMPASS.StatusDisplay "Waiting for AutoZero of the PMM..."
                           TimeDelay 15
                  ElseIf DAQ = 0 Then
                      'DaqType mode = 0: manual Set device
                      msg = msgbox("Press the VENT button on the chassis front panel and vent the reference
pressure device. Press OK when fully vented.",0+64+4096,"Manual Vent")
                      TimeDelay 3
                  End If

         ' Place the chassis into MEASURE mode to prepare for applying the pressure:
         cConfig.DUTPrs(1).IoSendCommand "OUTP:MODE MEASURE",True,0
         TimeDelay 2
         mode = cCOnfig.DUTPrs(1).IoSendCommand("OUTP:MODE?",False,0)
                  cdebug.LogStatus "Placed the chassis into MEASURE mode"

         cDebug.LogStatus "End of step <1100>."



'<currentteststep 1150 - pressure cycle 1 is done, run polyfit>
ElseIf cCOMPASS.CurrentTestStep = 1150 And iC = 1 Then
cCOMPASS.StatusDisplay "Calculating Adjustment..."
         cDebug.LogStatus "First Cycle is complete. Starting <step 1150> - run PolyFit"
    cDebug.LogStatus "#Data files: " & cCOMPASS.DataCollection.Count

     For i = 1 To cConfig.DUTPrs.Count
           pressurePoints = cCOMPASS.DataCollection(i).NumberofPressurePoints
           cDebug.LogStatus "DUT: " & i & " - Pressure Points: " & pressurePoints

           Redim Reference_Data(pressurePoints-1)
           Redim DUT_Raw_Data(pressurePoints-1)
           Redim C1_Corrected_Data(pressurePoints-1)

           'populate PMM A/F coefficients from memory:
           C0 = cCOMPASS.DataCollection(i).DUT.CalibrationCoefficient1
           C1 = cCOMPASS.DataCollection(i).DUT.CalibrationCoefficient2
           zOffset = cCOMPASS.DataCollection(i).DUT.CalibrationCoefficient3
                cDebug.LogStatus "As Found C0 coef: " & C0
                cDebug.LogStatus "As Found C1 coef: " & C1
                cDebug.LogStatus "As Found zOffset: " & zOffset

           ' Iterate through each pressure point
           cDebug.LogStatus "Calculating DUT_Raw(factory)_Data for each test point..."
           For j = 1 To pressurePoints
                ix = CInt(j)-1

                DUTPressure = cCOMPASS.DataCollection(i).DataPointRef(1, 1, 1, CInt(j)).DUTPressure
                                    Reference_Data(ix) = cCOMPASS.DataCollection(i).DataPointRef(1, 1, 1,
CInt(j)).RefPressure

                'Back out the As Found coefficients from each pressure point to determine the "Factory DUT
Pressure"...
                DUT_Raw_Data(ix)= (DUTPressure-C0-zOffset)/C1
                 cDebug.LogStatus "DUT_Raw_Data: " & DUT_Raw_Data(ix)

           Next

           Call Poly_Fit(Reference_Data, DUT_Raw_Data, res, 1)
           slope = res(0,1)
                cDebug.LogStatus "A/L slope: " & slope
           If slope = 0 Then
                new_C1 = 0
           Else
```

```
                new_C1 = 1 / slope
           End If
                cDebug.LogStatus "The new A/L C1 coef " & new_C1

           'Calculate the C1_corrected for each point:
           For j = 0 To pressurePoints - 1
                C1_Corrected_Data(CInt(j)) = DUT_Raw_Data(CInt(j)) * new_C1
                'cDebug.LogStatus "A/L Adjust C1_Corrected_Data(" & j & "): " & C1_Corrected_Data(CInt(j))
           Next

           Call Poly_Fit(Reference_Data, C1_Corrected_Data, res, 1)
           new_C0 = 0 - res(0,0)
                          cDebug.LogStatus "The new A/L C0 coef in the DUT unit of measure: " & new_C0

           'Need to convert the new_C0 back to kPa if this is a PM500 calibration.  If PM200, 600 then convert
the new_C0 back to psi...
           ModelTypeID = cConfig.DUTPrs(1).RangeMain.GetParent.Extra3
           DUTunitText = cCOnfig.DUTPrs(1).RangeMain.UnitFinalText
           DUTunit = cCOnfig.DUTPrs(1).RangeMain.UnitFinal
                          If ModelTypeID = 5 Then  'this is a PM500, therefore the C0 must go back to units of
kPa.
                                           cdebug.LogStatus "the Model Type ID is: " & ModelTypeID
                                           cdebug.LogStatus "the A/L new_C0 value is: " & new_C0 & DUTunitText
                                           new_C0bkcnvt =
cCOMPASS.UnitConversion(CDbl(new_C0),3,CInt(DUTunit),0)
                                           cdebug.LogStatus "the A/L new_C0 value converted back to kPa is: " &
new_C0bkcnvt
                                  Else
                                           'PM200 and PM600 always read the C0 in units of psi
                                           new_C0bkcnvt =
cCOMPASS.UnitConversion(CDbl(new_C0),9,CInt(DUTunit),0)
                                           cDebug.LogStatus "the A/L new_C0 value converted back to psi is: " &
new_C0bkcnvt
                                  End If


'Writing new data to the PMM.  Construct the new write commands, then loop through a Do-Loop
'to make sure that the new values were written.

           'Repopulate the A/F coefficients:
                          C0 = cCOMPASS.DataCollection(i).DUT.CalibrationCoefficient1
                          C1 = cCOMPASS.DataCollection(i).DUT.CalibrationCoefficient2
                          zOffset = cCOMPASS.DataCollection(i).DUT.CalibrationCoefficient3
                          calDate = cCOMPASS.DataCollection(i).DUT.LastCalDate
                                  cdebug.LogStatus "the As Found C0 coef is: " & C0
                                  cdebug.LogStatus "the As Found C1 coef is: " & C1

                          'Determine new calibration date:
                          new_date = ChassisCal_Date
                          cDebug.LogStatus "Calibration & Adjustment date: " & new_date

                          'Build new write commands:
                          PMMSlotAddress = cConfig.DUTPrs(i).RangeMain.GetParent.Extra2
                          cmd1 = "CAL:PRES" & PMMSlotAddress & ":DATA:VAL1?"
                          cmd2 = "CAL:PRES" & PMMSlotAddress & ":DATA:VAL2?"
                          cmd3 = "CAL:PRES" & PMMSlotAddress & ":ZERO:DATA?"
                          cmd4 = "CAL:PRES" & PMMSlotAddress & ":DATE?"

                          cmd5 = "CAL:PRES" & PMMSlotAddress & ":DATA:VAL1 " & new_C0bkcnvt
                          cmd6 = "CAL:PRES" & PMMSlotAddress & ":DATA:VAL2 " & new_C1
                          cmd7 = "CAL:PRES" & PMMSlotAddress & ":DATE " & new_date
                          cmd8 = "CAL:PRES" & PMMSlotAddress & ":ZERO:DATA 0"
                                  cdebug.LogStatus "cmd5 is: " & cmd5
                                  cdebug.LogStatus "cmd6 is: " & cmd6
                                  cdebug.LogStatus "cmd7 is: " & cmd7
                                  cdebug.LogStatus "cmd8 is: " & cmd8

                          'Write new coefficients to data file:
                          cCOMPASS.DataCollection(i).DUT.CalibrationCoefficient4 = new_C0
                          'Note: CalCoef4 is written in units of the DUT Test Unit (typically psi)
                          cCOMPASS.DataCollection(i).DUT.CalibrationCoefficient5 = new_C1
                          cCOMPASS.DataCollection(i).DUT.CalDueDate = CDate(new_date)

                          Do
                                  'Enable the CAL Mode:
                                  cConfig.DUTPrs(i).IoSendCommand "CAL:MODE 1",True
                                  TimeDelay 1
                                  mode = cconfig.DUTPrs(i).IoSendCommand ("CAL:MODE?",False)
                                           cdebug.LogStatus "Cal Mode is: " & mode

                                  'Write the new coefs:
                                  Call cConfig.DUTPrs(i).IoSendCommand(CStr(cmd5),True)
                                  TimeDelay 0.5
                                  Call cConfig.DUTPrs(i).IoSendCommand(CStr(cmd6),True)
                                  TimeDelay 0.5
```

```
                                        Call cConfig.DUTPrs(i).IoSendCommand(CStr(cmd7),True)
                                        TimeDelay 0.5
                                        Call cConfig.DUTPrs(i).IoSendCommand(CStr(cmd8),True)
                                        TimeDelay 0.5

                                        'Read back the written coefs:
                                        WtC0raw = cConfig.DUTPrs(i).IoSendCommand (CStr(cmd1),False)
                                        TimeDelay 0.5
                                        WtC1raw = cConfig.DUTPrs(i).IoSendCommand (CStr(cmd2),False)
                                        TimeDelay 0.5
                                        WtZ = cConfig.DUTPrs(1).IoSendCommand (CStr(cmd3),False)
                                        timedelay 0.5
                                                cdebug.LogStatus "WtC0raw value is: " & WtC0raw
                                                cdebug.LogStatus "WtC1raw value is: " & WtC1raw
                                                cdebug.LogStatus "Wtz value is: " & Wtz
                                        'Remove the leading coef identifier to attain only the value...
                                        WtC0 = qextract(WtC0raw,1,2,",")
                                        WtC1 = qextract(WtC1raw,1,2,",")
                                                cDebug.LogStatus "WtC0 value is: " & WtC0
                                                cDebug.LogStatus "WtC1 value is: " & WtC1
                                                cDebug.LogStatus "WtZ value is: " & WtZ

                                        'Check if the A/L is different from the A/F (it should be):
                                        C0Diff = C0 - WtC0
                                        C1Diff = C1 - WtC1
                                        ZoffDiff = zOffset - WtZ
                                                cDebug.LogStatus "C0 Difference: " & C0Diff
                                                cDebug.LogStatus "C1 Difference: " & C1Diff
                                                cDebug.LogStatus "Zoff Difference: " & ZoffDiff

                                        If cCOMPASS.SystemAbort Then Exit Function
                                        If C0Diff <> 0 And C1Diff <> 0 Then Exit Do
                                        cdebug.LogStatus "End of Loop."

                                cCOMPASS.StatusDisplay "Writing new coefficients..."
                                Loop
    Next

    Set obj = Nothing
    cdebug.LogStatus "Finished with CurrentTestStep 1150"


'<currentteststep 2000 - test is complete, A/L data has been collected. Save the coefs>

ElseIf cCOMPASS.CurrentTestStep = 2000 Then
cDebug.LogStatus "Test complete.  CurrentTestStep 2000.  Get final zOffset"

'Confirm the final zOffset is zero:
        PMMSlotAddress = cConfig.DUTPrs(1).RangeMain.GetParent.Extra2
        cmd9 = "CAL:PRES" & PMMSlotAddress & ":ZERO:DATA?"
        zOffset = cConfig.DUTPrs(1).IoSendCommand(CStr(cmd9), False)
        cDebug.LogStatus "Verification of A/L zOffset: " & zOffset

    ' Write the final zOffset value to the data file:
    cCOMPASS.DataCollection(1).DUT.CalibrationCoefficient6 = zOffset

'*************************************************************************************************************
*****

'This block of code is used to check the fit of the As Left corrections.  The key variable is named "TOLFACT".
'If TOLFACT = 1 then the As Left data must fit inside the 1 year tolerance specification.  If set to <1 then
'this acts as a form of guardbanding.

TOLFACT = .70   '70% gaurdband.  User definable.

Dim MessageBox1
Dim MessageBox2
                For i = 1 To cCOMPASS.DataCollection(1).NumberofPressurePoints
                        For j = 1 To cConfig.DUTPrs.Count
                             Ref = cCOMPASS.DataCollection(j).DataPointRef(1,1,2,CInt(i)).RefPressure
                             DUT = cCOMPASS.DataCollection(j).DataPointRef(1,1,2,CInt(i)).DUTPressure
                             Tol  = cCOMPASS.DataCollection(j).DataPointRef(1,1,2,CInt(i)).Tolerance
                                 ST = cCOMPASS.DataCollection(j).DataPointRef(1,1,2,CInt(i)).Status

                                        If abs(DUT-Ref) > Tol * TOLFACT Then
                                        FAIL = True

                                       'Give user the option of keeping or restoring the A/F calibration:
                                       cCOMPASS.StatusDisplay "Waiting for the user make a decision on the As Left
results..."
                                       MessageBox1 = msgbox("One or more of the As Left data points fall outside of
the allowed tolerance.  Click 'Yes' to accept the results, or click 'No' to cancel the adjustment and restore
the As Found coefficients.",4+48+4096,"Accept Results?")

                                       If MessageBox1 = 7 Then 'Note: 7 = yes, 6 = no
```

```
                                  Messagebox2 = msgbox("This will delete the newly determined As Found 
coefficients and return the PMM to how it was prior to the calibration.  Click 'Yes' to restore.",4+32+4096,
"Are you sure?")
                                  cdebug.LogStatus "The user chose to revert back to the A/F coefficients."
                                  End If
                                  If Messagebox2 = 6 Then  'Note: 6 = yes, I'm sure
                                  RestoreCal

                                  msgbox "The As Found coefficients have been restored."
                                  End If


                                  Exit For
                                   End If
                          Next
                  If fail = True Then Exit For

'END OF GUARD BANDING CODE.
'*****************************************************************************************************************
*****

        Next
        'Send a SAVE command for chassis with old firmware -  pre-v1.10 (6270A), v1.02 (8270A)...
        ID = cConfig.DUTPrs(1).IoSendCommand("*IDN?",False)
                cdebug.LogStatus "Response from ID command is: "& ID
        ModelRaw = qextract(ID,1,2,",")
                cdebug.LogStatus "The chassis model is: "& ModelRaw
        Model = Left(ModelRaw,4)
                cdebug.LogStatus "the simplified model nomenclature is: " & Model
        Firmware = qextract(ID,3,4,",")
        FirmwareCdbl = CDbl(Firmware)
                cdebug.LogStatus "The chassis firmware rev is: " & FirmwareCdbl

        PMMSlotAddress = cConfig.DUTPrs(1).RangeMain.GetParent.Extra2
        cmd10 = "CAL:PRES" & PMMSlotAddress & ":SAVE"

        If Model = "6270" And FirmwareCdbl < 1.10  Then
                Call cConfig.DUTPrs(1).IoSendCommand(CStr(cmd10),True)
                TimeDelay 0.5
                cdebug.LogStatus "the detected model was 6270A with an older ver of firmware."
                cdebug.LogStatus "Data saved to PMM."
        ElseIf Model = "8270" And FirmwareCdbl < 1.02 Then
                Call cConfig.DUTPrs(1).IoSendCommand(CStr(cmd10),True)
                TimeDelay 0.5
                cdebug.LogStatus "the detected model was 8270A with an older ver of firmware."
                cdebug.LogStatus "Data saved to PMM."
        ElseIf Model = "8370" And FirmwareCdbl < 1.02 Then
                Call cConfig.DUTPrs(1).IoSendCommand(CStr(cmd10),True)
                TimeDelay 0.5
                cdebug.LogStatus "the detected model was 8370A with an older ver of firmware."
                cdebug.LogStatus "Data saved to PMM."
    End If

End If
End Function
```

ChassisCal_Absolute:

```
'*****************************************************************************************************
'Used with ChassisCal_Master to calibrate and adjust Absolute mode PMMs.  Applicable for 6270A / 8270A / 8370A.
'Called by the master test macro "ChassisCal_Master".  The Absolute mode calibrations do not zero the PMM prior
'To taking A/F Data.

'*****************************************************************************************************

Function ChassisCal_Absolute(iT, iL, iC, iP, cTest, cConfig)
Dim DUT_Raw_Data()
Dim Reference_Data()
Dim C1_Corrected_Data()
Dim res(4,7)

C0 = 0
C1 = 0
zOffset = 0


'<currentteststep 1010 - record AF coefs.>
If cCOMPASS.CurrentTestStep = 1010 Then ' Test Definition data files were created for each DUT.
    cDebug.LogStatus "Starting <step 1010>  "

        'Ensure the Chassis is in Absolute mode:
        cConfig.DUTPrs(1).IoSendCommand "SENS:MODE ABS",True

        DAQ = cCOMPASS.cConfig.SetPrs(1).RangeMain.GetParent.DaqType
        cdebug.LogStatus "Data Acquisition Type for SetPrs is: " & DAQ

    If DAQ <> 0 Then
                'Vent both the Reference and 6270A.
                cCOMPASS.cConfig.SetPrs(1).IoSetOutput 0,0,1
                cCOMPASS.cConfig.DUTPrs(1).IoSetOutput 0,0,1
                        cDebug.LogStatus "Sent ioSetOutput 0,0,1 to vent the Ref Device."
                        cDebug.LogStatus "Sent ioSetOutput 0,0,1 to vent the 6270A."
                cCOMPASS.StatusDisplay "Venting the pressure..."
                TimeDelay 3
        ElseIf DAQ = 0 Then
            'DaqType mode = 0: manual Set device
            msg = msgbox("Press the VENT button on the 6270A front panel and vent the reference pressure
device. Press OK when fully vented.",0+64+4096,"Manual Vent")
            TimeDelay 5
        End If


    cDebug.LogStatus "Capture As Found Coeffs..."
        PMMSlotAddress = cConfig.DUTPrs(1).RangeMain.GetParent.Extra2
                cDebug.LogStatus "The PMM slot address is: " & PMMSlotAddress

        cmd1 = "CAL:PRES" & PMMSlotAddress & ":DATA:VAL1?"
        cmd2 = "CAL:PRES" & PMMSlotAddress & ":DATA:VAL2?"
        cmd3 = "CAL:PRES" & PMMSlotAddress & ":ZERO:DATA?"
        cmd4 = "CAL:PRES" & PMMSlotAddress & ":DATE?"

'Identify the unit of measure being used for the test and the PMM Type:
DUTunit = cConfig.DUTPrs(1).RangeMain.UnitFinal
DUTunitText = cConfig.DUTPrs(1).RangeMain.UnitFinalText
ModelTypeID = cConfig.DUTPrs(1).RangeMain.GetParent.Extra3

        ' Read A/F C0 from PMM; make unit conversions, write to calcoef1:
        C0raw = cConfig.DUTPrs(1).IoSendCommand(CStr(cmd1), False)
                cDebug.LogStatus "the raw C0 string is: " & C0raw
        C0 = qextract(C0raw,1,2,",")
                If ModelTypeID = 5 Then  'this is a PM500, therefore the C0 is read in units of kPa
                        cdebug.LogStatus "the A/F C0 value in kPa is: " & C0
                C0cnvt = cCOMPASS.UnitConversion(CDbl(C0),CInt(DUTunit),3,0)
                        cdebug.LogStatus "the A/F C0 value converted to the DUT unit of measure is: " &
C0cnvt & DUTunitText
                Else
                'PM200 and PM600 always read the C0 in units of psi
                cDebug.LogStatus "The C0 value before conversion is: " & C0
                C0cnvt = cCOMPASS.UnitConversion(CDbl(C0),CInt(DUTunit),9,0)
                        cdebug.LogStatus "the A/F C0 value converted to the DUT unit of measure is: " &
C0cnvt & DUTunitText
                End If
        cCOMPASS.DataCollection(1).DUT.CalibrationCoefficient1 = C0cnvt


        ' Read A/F C1 from DUT; write to calcoef2
        C1raw = cConfig.DUTPrs(1).IoSendCommand(CStr(cmd2), False)
        C1 = qextract(C1raw,1,2,",")
        cCOMPASS.DataCollection(1).DUT.CalibrationCoefficient2 = C1
                cDebug.LogStatus "As Found C1 coefficient: " & C1
        ' Read A/F zOffset from PMM, write to calcoef3:
        ' Note: zOffset is displayed on front panel and queried in the same unit of measure as the chassis.
        zOffset = cConfig.DUTPrs(1).IoSendCommand(CStr(cmd3), False)
        cCOMPASS.DataCollection(1).DUT.CalibrationCoefficient3 = zOffset
                cDebug.LogStatus "As Found zOffset is: " & zOffset
                cDebug.LogStatus "the zOffset value is in units of: " &
cConfig.DUTPrs(1).RangeMain.UnitFinalText

        ' Read the last saved cal edit date:
        calDate = cConfig.DUTPrs(1).IoSendCommand(CStr(cmd4), False)
        cCOMPASS.DataCollection(1).DUT.LastCalDate = calDate
                cDebug.LogStatus "As Found last cal date: " & calDate


        ' Place the 6270A into MEASURE mode to prepare for applying the pressure:
        cConfig.DUTPrs(1).IoSendCommand "OUTP:MODE MEASURE",True,0
        TimeDelay 2
        mode = cConfig.DUTPrs(1).IoSendCommand("OUTP:MODE?",False,0)
                cdebug.LogStatus "Measurement mode: " & mode
                cdebug.LogStatus "Placed the 6270A into MEASURE mode"

        cdebug.LogStatus "<End of CurrentTestStep 1010>"

'<The test definition will now run the first cycle of points>
'<Continue with pressure cycle 1 (as-found data)>


'<currentteststep 1150 - pressure cycle 1 is done, run polyfit>
ElseIf cCOMPASS.CurrentTestStep = 1150 And iC = 1 Then
cCOMPASS.StatusDisplay "Calculating Adjustment..."
        cDebug.LogStatus "First Cycle is complete. Starting <step 1150> - run PolyFit"
    cDebug.LogStatus "#Data files: " & cCOMPASS.DataCollection.Count

     For i = 1 To cConfig.DUTPrs.Count
          pressurePoints = cCOMPASS.DataCollection(i).NumberofPressurePoints
          cDebug.LogStatus "DUT: " & i & " - Pressure Points: " & pressurePoints

          Redim Reference_Data(pressurePoints-1)
          Redim DUT_Raw_Data(pressurePoints-1)
          Redim C1_Corrected_Data(pressurePoints-1)

          'populate PMM A/F coefficients from memory:
          C0 = cCOMPASS.DataCollection(i).DUT.CalibrationCoefficient1
          C1 = cCOMPASS.DataCollection(i).DUT.CalibrationCoefficient2
          zOffset = cCOMPASS.DataCollection(i).DUT.CalibrationCoefficient3
               cDebug.LogStatus "As Found C0 coef: " & C0
               cDebug.LogStatus "As Found C1 coef: " & C1
               cDebug.LogStatus "As Found zOffset: " & zOffset

          'Iterate through each pressure point:
                       cDebug.LogStatus "Calculating DUT_Raw(factory)_Data for each test point..."
          For j = 1 To pressurePoints
               ix = CInt(j)-1

               DUTPressure = cCOMPASS.DataCollection(i).DataPointRef(1, 1, 1, CInt(j)).DUTPressure
                                Reference_Data(ix) = cCOMPASS.DataCollection(i).DataPointRef(1, 1, 1,
CInt(j)).RefPressure

               'Back out the As Found coefficients from each pressure point to determine the "Factory DUT
Pressure"...
               DUT_Raw_Data(ix)= (DUTPressure-C0-zOffset)/C1
                cDebug.LogStatus "DUT_Raw_Data: " & DUT_Raw_Data(ix)

          Next

          Call Poly_Fit(Reference_Data, DUT_Raw_Data, res, 1)
          slope= res(0,1)
               cDebug.LogStatus "A/L slope: " & slope
          If slope = 0 Then
               new_C1 = 0
          Else
               new_C1 = 1 / slope
          End If
               cDebug.LogStatus "The new A/L C1 coef " & new_C1

          'Calculate the C1_corrected for each point:
          For j = 0 To pressurePoints - 1
               C1_Corrected_Data(CInt(j)) = DUT_Raw_Data(CInt(j)) * new_C1
               'cDebug.LogStatus "A/L Adjust C1_Corrected_Data(" & j & "): " & C1_Corrected_Data(CInt(j))
          Next


          Call Poly_Fit(Reference_Data, C1_Corrected_Data, res, 1)
          new_C0 = 0 - res(0,0)
```

```
                          cDebug.LogStatus "The new A/L C0 coef in the DUT unit of measure: " & new_C0

           'Need to convert the new_C0 back to kPa if this is a PM500 calibration.  If PM200, 600 then convert
the new_C0 back to psi...
           DUTunit = cCOnfig.DUTPrs(1).RangeMain.UnitFinal
           DUTunitText = cCOnfig.DUTPrs(1).RangeMain.UnitFinalText
           ModelTypeID = cConfig.DUTPrs(1).RangeMain.GetParent.Extra3

                          If ModelTypeID = 5 Then  'this is a PM500, therefore the new_C0 must go back to units
of kPa to be written to chassis.
                                          cdebug.LogStatus "the Model Type ID is: " & ModelTypeID
                                          cdebug.LogStatus "the A/L new_C0 value is: " & new_C0 & DUTunitText
                                          new_C0bkcnvt =
cCOMPASS.UnitConversion(CDbl(new_C0),3,CInt(DUTunit),0)
                                          cdebug.LogStatus "the A/L new_C0 value converted back to kPa is: " &
new_C0bkcnvt
                                  Else
                                          'PM200 and PM600 always read the C0 in units of psi
                                          new_C0bkcnvt =
cCOMPASS.UnitConversion(CDbl(new_C0),9,CInt(DUTunit),0)
                                          cDebug.LogStatus "the A/L new_C0 value converted to psi is: " &
new_C0bkcnvt
                                  End If


'Writing new data to the PMM.  Construct the new write commands, then loop through a Do-Loop
'to make sure that the new values were written...

            'Repopulate the A/F coefficients:
                          C0 = cCOMPASS.DataCollection(i).DUT.CalibrationCoefficient1
                          C1 = cCOMPASS.DataCollection(i).DUT.CalibrationCoefficient2
                          zOffset = cCOMPASS.DataCollection(i).DUT.CalibrationCoefficient3
                          calDate = cCOMPASS.DataCollection(i).DUT.LastCalDate
                                  cdebug.LogStatus "the As Found C0 coef is: " & C0
                                  cdebug.LogStatus "the As Found C1 coef is: " & C1

                          'Determine new calibration date:
                          new_date = ChassisCal_Date
                          cDebug.LogStatus "Calibration & Adjustment date: " & new_date

                          'Build new write commands:
                          PMMSlotAddress = cConfig.DUTPrs(i).RangeMain.GetParent.Extra2
                          cmd1 = "CAL:PRES" & PMMSlotAddress & ":DATA:VAL1?"
                          cmd2 = "CAL:PRES" & PMMSlotAddress & ":DATA:VAL2?"
                          cmd3 = "CAL:PRES" & PMMSlotAddress & ":ZERO:DATA?"
                          cmd4 = "CAL:PRES" & PMMSlotAddress & ":DATE?"

                          cmd5 = "CAL:PRES" & PMMSlotAddress & ":DATA:VAL1 " & new_C0bkcnvt
                          cmd6 = "CAL:PRES" & PMMSlotAddress & ":DATA:VAL2 " & new_C1
                          cmd7 = "CAL:PRES" & PMMSlotAddress & ":ZERO:DATA 0"
                          cmd8 = "CAL:PRES" & PMMSlotAddress & ":DATE " & new_date

                                  cdebug.LogStatus "cmd5 is: " & cmd5
                                  cdebug.LogStatus "cmd6 is: " & cmd6
                                  cdebug.LogStatus "cmd7 is: " & cmd7
                                  cdebug.LogStatus "cmd8 is: " & cmd8

                          'Write new coefficients to data file:
                          cCOMPASS.DataCollection(i).DUT.CalibrationCoefficient4 = new_C0
                          'Note: CalCoef4 is written in units of the DUT Test Unit (typically psi)
                          cCOMPASS.DataCollection(i).DUT.CalibrationCoefficient5 = new_C1
                          cCOMPASS.DataCollection(i).DUT.CalDueDate = CDate(new_date)

                          Do
                                  'Enable the CAL Mode:
                                  cConfig.DUTPrs(i).IoSendCommand "CAL:MODE 1",True
                                  TimeDelay 1
                                  mode = cconfig.DUTPrs(i).IoSendCommand ("CAL:MODE?",False)
                                          cdebug.LogStatus "Cal Mode is: " & mode

                                  'Write the new coefs:
                                  Call cConfig.DUTPrs(i).IoSendCommand(CStr(cmd5),True)
                                  TimeDelay 0.5
                                  Call cConfig.DUTPrs(i).IoSendCommand(CStr(cmd6),True)
                                  TimeDelay 0.5
                                  Call cConfig.DUTPrs(i).IoSendCommand(CStr(cmd7),True)
                                  TimeDelay 0.5
                                  Call cConfig.DUTPrs(i).IoSendCommand(CStr(cmd8),True)
                                  TimeDelay 0.5

                                  'Read back the written coefs:
                                  WtC0raw = cConfig.DUTPrs(i).IoSendCommand (CStr(cmd1),False)
                                  TimeDelay 0.5
                                  WtC1raw = cConfig.DUTPrs(i).IoSendCommand (CStr(cmd2),False)
                                  TimeDelay 0.5
```

```
                                   WtZ = cConfig.DUTPrs(1).IoSendCommand (CStr(cmd3),False)
                                   TimeDelay 0.5
                                            cdebug.LogStatus "WtC0raw value is: " & WtC0raw
                                            cdebug.LogStatus "WtC1raw value is: " & WtC1raw
                                            cdebug.LogStatus "Wtz value is: " & Wtz
                                   'Remove the leading coef identifier to attain only the value...
                                   WtC0 = qextract(WtC0raw,1,2,",")
                                   WtC1 = qextract(WtC1raw,1,2,",")
                                            cDebug.LogStatus "WtC0 value is: " & WtC0
                                            cDebug.LogStatus "WtC1 value is: " & WtC1
                                            cDebug.LogStatus "WtZ value is: " & WtZ

                                   'Check if the A/L is different from the A/F (it should be):
                                   C0Diff = C0 - WtC0
                                   C1Diff = C1 - WtC1
                                   ZoffDiff = zOffset - WtZ
                                            cDebug.LogStatus "C0 Difference: " & C0Diff
                                            cDebug.LogStatus "C1 Difference: " & C1Diff
                                            cDebug.LogStatus "Zoff Difference: " & ZoffDiff

                                   If cCOMPASS.SystemAbort Then Exit Function
                                   If C0Diff <> 0 And C1Diff <> 0 Then Exit Do
                                   cdebug.LogStatus "End of Loop."

                          cCOMPASS.StatusDisplay "Writing new coefficients..."
                          Loop
     Next

     Set obj = Nothing
     cdebug.LogStatus "Finished with CurrentTestStep 1150"


'<currentteststep 2000 - test is complete, AL data has been collected. Save the coefs>

ElseIf cCOMPASS.CurrentTestStep = 2000 Then
cDebug.LogStatus "Test complete.  CurrentTestStep 2000.  Get final zOffset"

        'Confirm the final zOffset is zero:
        PMMSlotAddress = cConfig.DUTPrs(1).RangeMain.GetParent.Extra2
        cmd9 = "CAL:PRES" & PMMSlotAddress & ":ZERO:DATA?"
        zOffset = cConfig.DUTPrs(1).IoSendCommand(CStr(cmd9), False)
        cDebug.LogStatus "Verification of A/L zOffset: " & zOffset

    ' Write the final zOffset value to the data file:
    cCOMPASS.DataCollection(1).DUT.CalibrationCoefficient6 = zOffset

'***********************************************************************************************************
*****
'This block of code is used to check the fit of the As Left corrections.  The key variable is named "TOLFACT".
'If TOLFACT = 1 then the As Left data must fit inside the 1 year tolerance specification.  If set to <1 then
'this acts as a form of guardbanding.

TOLFACT = .70   '70% gaurdband.  User definable.

Dim MessageBox1
Dim MessageBox2
                For i = 1 To cCOMPASS.DataCollection(1).NumberofPressurePoints
                        For j = 1 To cConfig.DUTPrs.Count
                             Ref = cCOMPASS.DataCollection(j).DataPointRef(1,1,2,CInt(i)).RefPressure
                             DUT = cCOMPASS.DataCollection(j).DataPointRef(1,1,2,CInt(i)).DUTPressure
                             Tol  = cCOMPASS.DataCollection(j).DataPointRef(1,1,2,CInt(i)).Tolerance
                                 ST = cCOMPASS.DataCollection(j).DataPointRef(1,1,2,CInt(i)).Status

                                If abs(DUT-Ref) > Tol * TOLFACT Then
                                FAIL = True

                               'Give user the option of keeping or restoring the A/F calibration:
                               cCOMPASS.StatusDisplay "Waiting for the user make a decision on the As Left
results..."
                               MessageBox1 = msgbox("One or more of the As Left data points fall outside of
the allowed tolerance.  Click 'Yes' to accept the results, or click 'No' to cancel the adjustment and restore
the As Found coefficients.",4+48+4096,"Accept Results?")

                               If MessageBox1 = 7 Then 'Note: 7 = yes, 6 = no
                               Messagebox2 = msgbox("This will delete the newly determined As Found
coefficients and return the PMM to how it was prior to the calibration.  Click 'Yes' to restore.",4+32+4096,
"Are you sure?")
                               cdebug.LogStatus "The user chose to revert back to the A/F coefficients."
                               End If
                               If Messagebox2 = 6 Then  'Note: 6 = yes, I'm sure
                               RestoreCal

                               msgbox "The As Found coefficients have been restored."
                               End If
```

```
                                  Exit For
                                  End If
                          Next
                 If fail = True Then Exit For

'END OF GUARD BANDING CODE.
'**************************************************************************************************************
*****

        Next
        'Send a SAVE command for chassis with old firmware -  pre-v1.10 (6270A), v1.02 (8270A)...
        ID = cConfig.DUTPrs(1).IoSendCommand("*IDN?",False)
                cdebug.LogStatus "Response from ID command is: "& ID
        ModelRaw = qextract(ID,1,2,",")
                cdebug.LogStatus "The chassis model is: "& ModelRaw
        Model = Left(ModelRaw,4)
                cdebug.LogStatus "the simplified model nomenclature is: " & Model
        Firmware = qextract(ID,3,4,",")
        FirmwareCdbl = CDbl(Firmware)
                cdebug.LogStatus "The chassis firmware rev is: " & FirmwareCdbl

        PMMSlotAddress = cConfig.DUTPrs(1).RangeMain.GetParent.Extra2
        cmd10 = "CAL:PRES" & PMMSlotAddress & ":SAVE"

        If Model = "6270" And FirmwareCdbl < 1.10  Then
                Call cConfig.DUTPrs(1).IoSendCommand(CStr(cmd10),True)
                TimeDelay 0.5
                cdebug.LogStatus "the detected model was 6270A with an older ver of firmware."
                cdebug.LogStatus "Data saved to PMM."
        ElseIf Model = "8270" And FirmwareCdbl < 1.02 Then
                Call cConfig.DUTPrs(1).IoSendCommand(CStr(cmd10),True)
                TimeDelay 0.5
                cdebug.LogStatus "the detected model was 8270A with an older ver of firmware."
                cdebug.LogStatus "Data saved to PMM."
        ElseIf Model = "8370" And FirmwareCdbl < 1.02 Then
                Call cConfig.DUTPrs(1).IoSendCommand(CStr(cmd10),True)
                TimeDelay 0.5
                cdebug.LogStatus "the detected model was 8370A with an older ver of firmware."
                cdebug.LogStatus "Data saved to PMM."
    End If

End If
End Function
```