UCF Test Executive Configuration Notes v013023
Richard Todd
Test Engineering Services
Requires libTeSequencer.(so | dll) v012623 or later.
Requires UniConsole(.exe) v012623 or later.
Requires StdIOChannelSystem(.so | dll) v012623 or later.

1. Description

This document outlines the necessary steps to configure the UCF test executive framework on startup.

2. Terminology

Term	Definition
UCF	The UniConsole Framework
Management Console	The tab/text based console that is used to monitor and control all application functionality
Test Executive	The user interface used to automatically sequence all tests and save measurement data to a local database
Go-nogo	A test sequence initiated by the test executive will stop on the first measurement failure

3. Configuration Settings

The following test executive startup settings can be configured using system variables. These settings are optional.

a) Management Console GUI interface size and location

b) Test Executive GUI interface size and location

c) The initial test mode (Go-nogo)

d) The manufacturer's name and location of the test station

e) Test specifications, parameters and limits for each test profile

f) The number of units to be tested concurrently

g) Manual UUT serial number verification

3.2. System variables

System variables are stored in local or global sqlite databases and are configured using the ‘DB Browser for Sqlite’ application.

Locations

Database Type	Database Location	Platform
Local	*~/ucfroot/opt/<Profile Family Name>/ExtSystemIO.sdb*	Linux
	C:\ucfroot\opt\<Profile Family Name>\ExtSystemIO.sdb	Windows
Global	*~/ucfroot/bin/<Release Name>/IOConfig/BaseSystemIO.sdb*	Linux
	C:\ucfroot\bin\<Release Name>\IOConfig\BaseSystemIO.sdb	Windows

a) Local database values are unique to a specific profile family and will override values in global databases.

b) Global database values will be used for all profiles that do not have overriding values in a local database.

4. User Interface Configuration

The Management Console and Test Executive user interface can be optionally configured using system variables.

- Management Console window size & location
- Test Executive window size & location
- Initial test mode (Go-nogo enable)

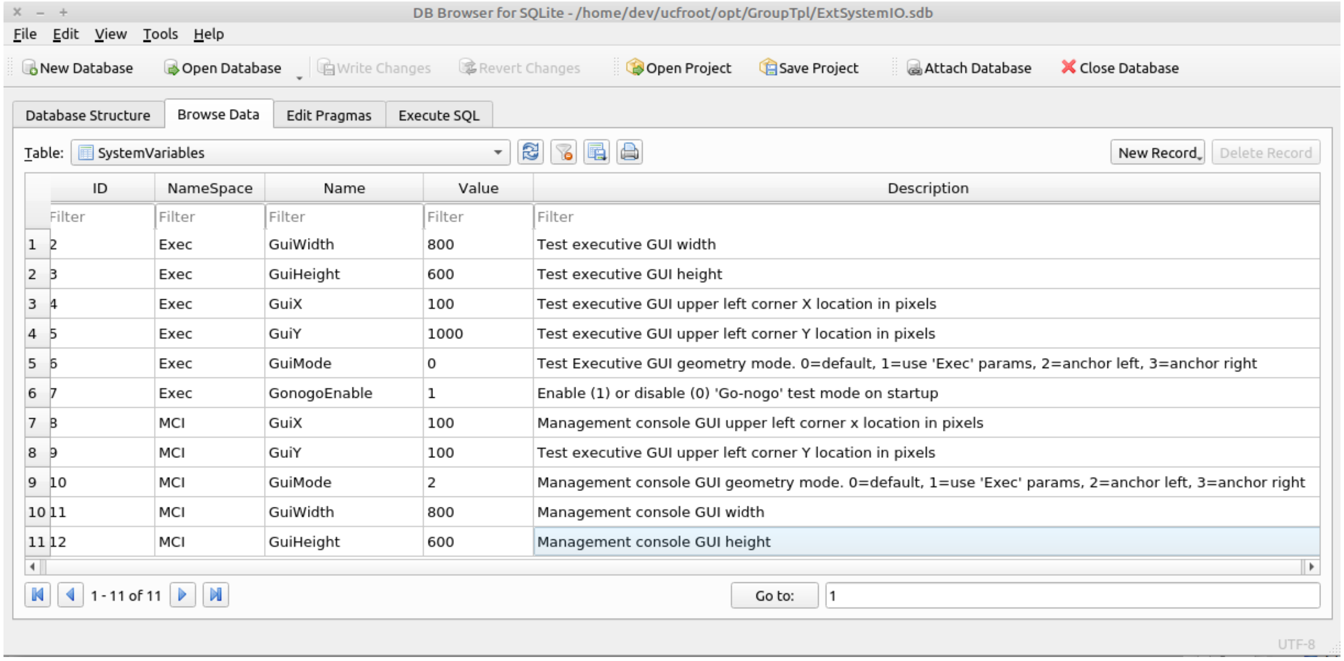


Table: SystemVariables

	ID	NameSpace	Name	Value	Description
	Filter	Filter	Filter	Filter	Filter
1	2	Exec	GuiWidth	800	Test executive GUI width
2	3	Exec	GuiHeight	600	Test executive GUI height
3	4	Exec	GuiX	100	Test executive GUI upper left corner X location in pixels
4	5	Exec	GuiY	1000	Test executive GUI upper left corner Y location in pixels
5	6	Exec	GuiMode	0	Test Executive GUI geometry mode. 0=default, 1=use 'Exec' params, 2=anchor left, 3=anchor right
6	7	Exec	GonogoEnable	1	Enable (1) or disable (0) 'Go-nogo' test mode on startup
7	8	MCI	GuiX	100	Management console GUI upper left corner x location in pixels
8	9	MCI	GuiY	100	Test executive GUI upper left corner Y location in pixels
9	10	MCI	GuiMode	2	Management console GUI geometry mode. 0=default, 1=use 'Exec' params, 2=anchor left, 3=anchor right
10	11	MCI	GuiWidth	800	Management console GUI width
11	12	MCI	GuiHeight	600	Management console GUI height

1 - 11 of 11

Go to: 1

Where

‘Exec::GuiMode’ specifies the display mode of the test executive user interface.

0=default
1=use ‘Exec’ system variables to set geometry
2=anchor left
3=anchor right

‘MCI::GuiMode’ specifies the display mode of the management console user interface.

0=default
1=use ‘Exec’ system variables to set geometry
2=anchor left
3=anchor right

‘Exec::GuiMode’ specifies the execution mode of the test executive on startup.

0=Do not stop on first test failure
1=Stop on first test failure

5. Test Specification Configuration Process

The test executive is designed to use test limits from a version controlled test specification. Each parameter saved in the database is traceable to

- The test specification used to determine pass/fail status
- The manufacturers name and physical location
- The test station used to save the parameters

Test specification information is stored in local sqlite database and is configured using the ‘DB Browser for Sqlite’ application.

Database Type	Database Location	Platform
Manufacturing ID	*/var/lib/ucf/Static.sdb*	Linux
	C:\ucfroot\var\lib\Static.sdb	Windows
Test Specifications	*/var/lib/ucf/TestLimits.sdb*	Linux
	C:\ucfroot\var\lib\TestLimits.sdb	Windows

5.1. Manufacturing ID

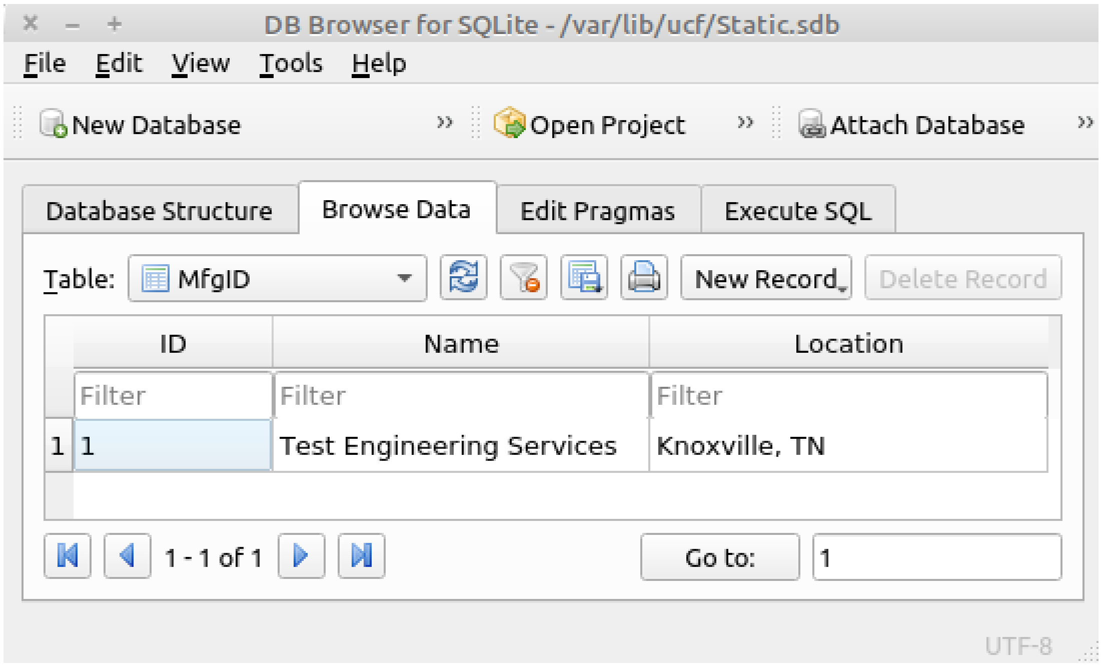


Table: MfgID

	ID	Name	Location
1	1	Test Engineering Services	Knoxville, TN

Go to: 1

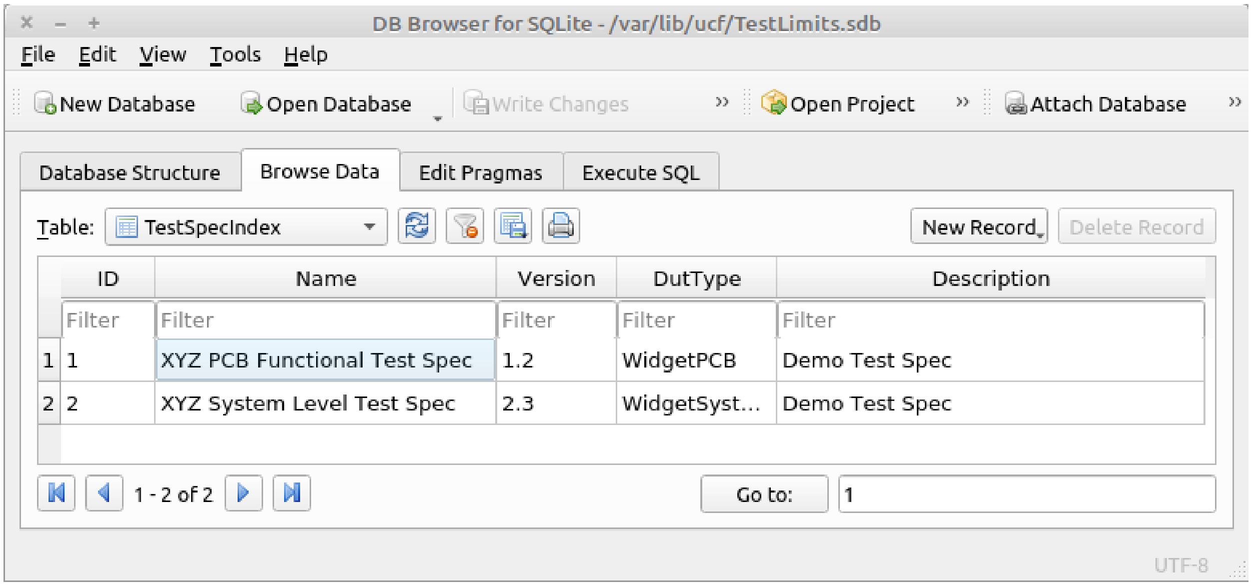
a) ‘Name’ is the name of the manufacturing company.

b) ‘Location’ is the location of the manufacturing company.

c) Only the first record is used. All additional entries will be ignored.

5.2. Test Specification(s)

Any number of test specifications can be set in the TestLimits::TestSpecIndex table.



	ID	Name	Version	DutType	Description
	Filter	Filter	Filter	Filter	Filter
1	1	XYZ PCB Functional Test Spec	1.2	WidgetPCB	Demo Test Spec
2	2	XYZ System Level Test Spec	2.3	WidgetSyst...	Demo Test Spec

a) ‘ID’ is the numeric identifier used to reference each test specification

b) ‘Name’ is the name of the test specification

c) ‘Version’ is the version of the test specification. Multiple versions can be specified

d) ‘DutType’ is the name/type of UUT to be tested

e) ‘Description’ is a brief description of the test specification

5.3. Test Profile Maps(s)

The test profile map table specifies which test specification will be used for each UCF profile.

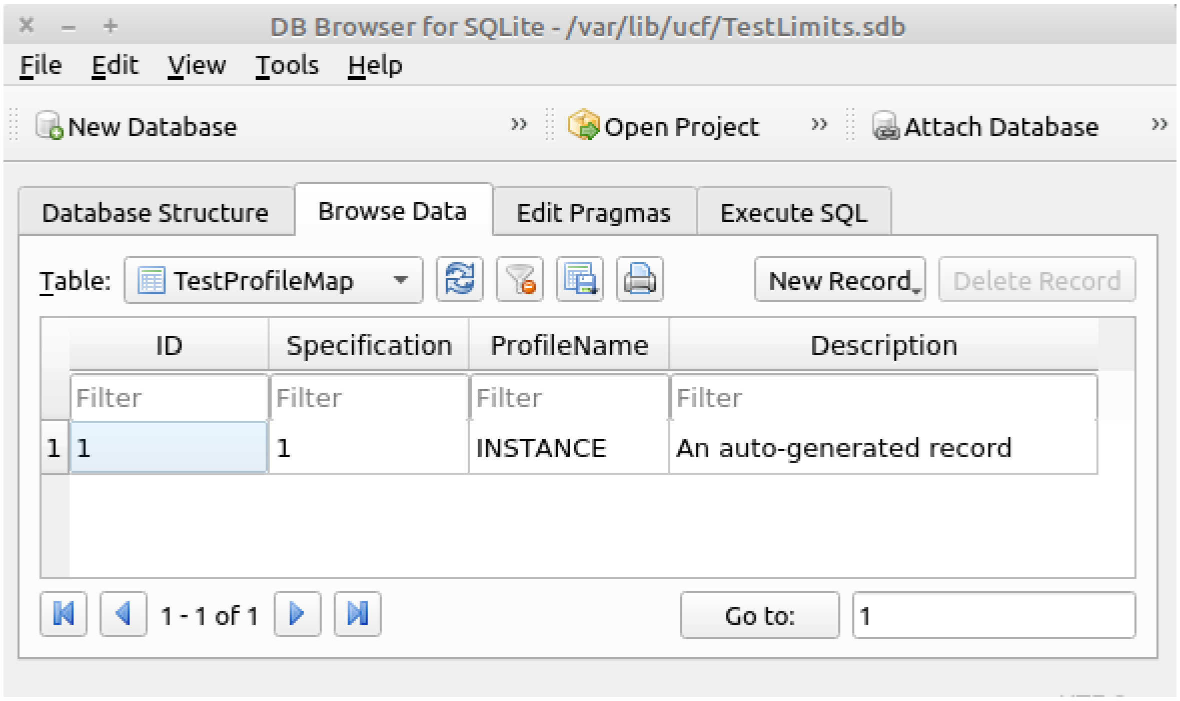


Table: TestProfileMap

	ID	Specification	ProfileName	Description
	Filter	Filter	Filter	Filter
1	1	1	INSTANCE	An auto-generated record

1 - 1 of 1

Go to: 1

a) Specification' corresponds to a test specification ID defined in TestLimits::TestSpecIndex

b) 'ProfileName' is the last element in the file path specified when starting the UniConsole framework.

Example UCF startup command:

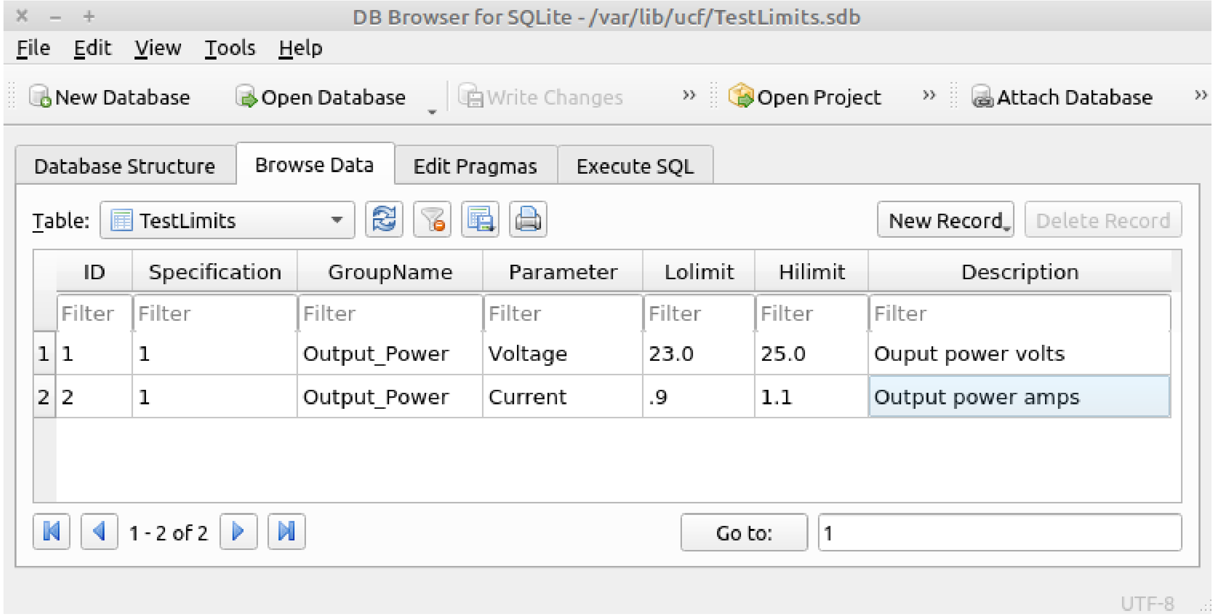
'./UniConsole(.exe) -p /home/dev/ucfroot/opt/GroupTpl/**INSTANCE**'

Where **INSTANCE** is the profile name

6. Test Limits

The test profile map table specifies which test specification will be used for each framework profile.

This table will be automatically populated the first time the test sequence is executed and will use the the default values specified in the test program. Subsequent test sequence executions **will not** modify the test limits.



	ID	Specification	GroupName	Parameter	Lolimit	Hilimit	Description
	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	1	1	Output_Power	Voltage	23.0	25.0	Ouput power volts
2	2	1	Output_Power	Current	.9	1.1	Output power amps

a) Specification’ corresponds to a test specification ID defined in TestLimits::TestSpecIndex.

b) ‘GroupName' is a category or namespace.

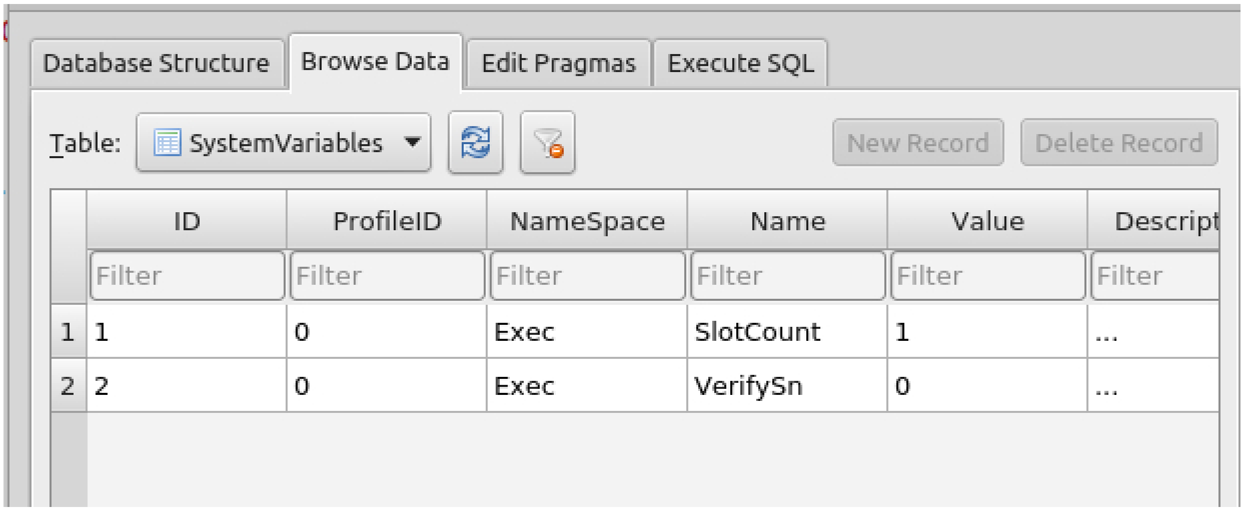
c) ‘Parameter’ is the name of a specific measurement.

d) ‘Lolimit’ is the lower pass/fail threshold. Passing values are >= ‘Lolimit’.

e) ‘Hilimit’ is the upper pass/fail threshold. Passing values are <= 'Hilimit.

7. Slot Count & Verify Serial Number

The framework is capable of testing 1-16 UUT's concurrently and can require that a serial number be manually entered at the start of a test sequence. These values are optional and will default to ‘1’.



	ID	ProfileID	NameSpace	Name	Value	Descript
	Filter	Filter	Filter	Filter	Filter	Filter
1	1	0	Exec	SlotCount	1	...
2	2	0	Exec	VerifySn	0	...

a) ‘ProfileID’ is not used.

b) ‘NameSpace’ = ‘Exec’

c) ‘Name’ = ‘SlotCount’ | ‘VerifySn’

d) ‘SlotCount’ is the number of units to be tested concurrently (1-16).

e) ‘VerifySn’ enable/disables manual input of a serial number at the start of a test sequence.

See section 3.2 for additional system variable information