

**MultiTester for MT4
Version 1.1**

www.MT4-MultiTester.com

What is the MultiTester ?

The MultiTester is a set of software tools to perform simultaneous multiple tester optimization runs, conveniently controlled by a single MT4 master terminal. This allows the full use of all available CPU resources to speed up the optimization process. In addition to using the local CPU cores also a second computer in a local network can be used to further speed up the optimization. For example by use of two octacore computers in a home network, the time for a long optimization - lasting half an hour in a standard MT4 set up - can drop to as little as 2 minutes.

How does it work ?

In order to get all CPU cores to work the MultiTester splits the optimization task into subtasks, one for each CPU core in your configuration. Then for each core a slave MT4 terminal is started for parallel processing of these sub task. After all slave terminals have completed their processing the results are automatically merged into combined optimization report.

Main Features:

- Fully integrated into the MT4 user interface.
- Allows an unrestricted number of parallel optimizations (however it is not useful to exceed the number of CPU cores).
- Supports optimizations on one remote computer in a local network
- Supports two user interface modes:
 - o For EA programmers an optimization and EA software change is done in the exactly same workflow as with the standard MT4 installation.
 - o If you don't have your EA's source code an alternative MultiTester_control EA can be used
- Supporting Scripts to set up and manage your slave tester farm are provided
- Support for spread changing tools is provided to allow tests with altered spreads
- Enhanced optimization report ready for spreadsheet evaluation, with all EA settings and information about the spread of the performed tests

Detailed Instruction

The MT4-MultiTester must be installed into the MT4 installation you are using to test or develop your EA. Let's call this MT4 instance your MT4 master instance. It is assumed here, that your EA is fully working on this MT4 installation:

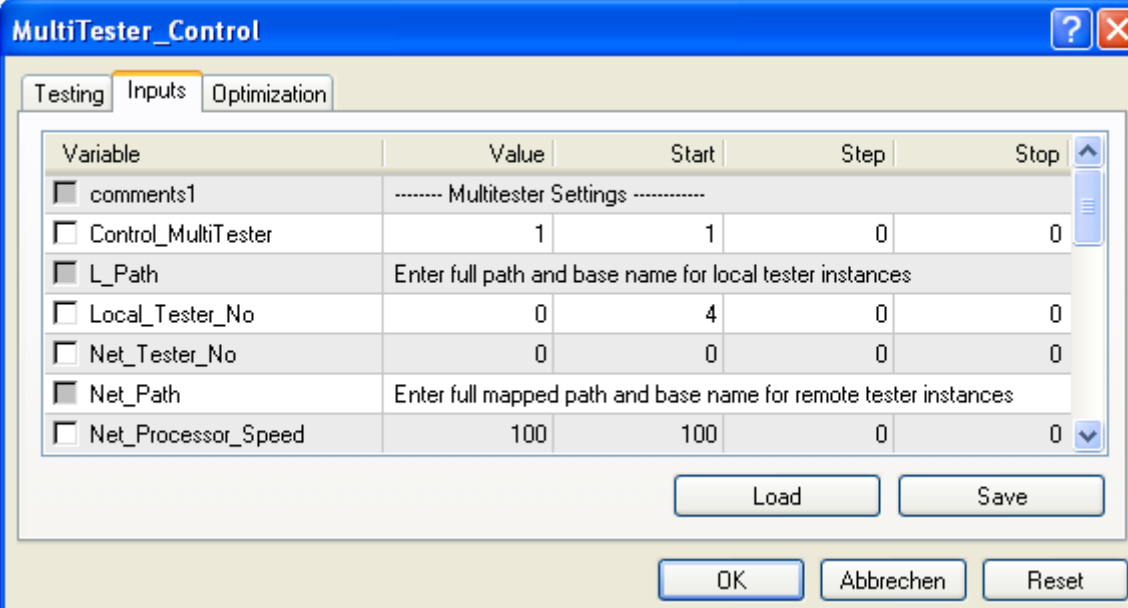
- All necessary histories are loaded, all libraries, indicators etc. you need for your EA are there.
- If you test or optimize your EA on this installation it does work.

Now simply copy the files of the MultiTester_install_v1.1.zip into the root of your tester. You should then find the following files in your MT4 master directories:

Directory Name	Filename
\	MultiTester_Remote.bat (service for remote computer) guardian.exe (copy protection) README_GUARDIAN.txt (read me for copy protection) MT4-MultiTester_Manual.pdf (this manual)
\experts	MultiTesterControl.mq4, MultiTesterControl.ex4 MACD_Sample_M.mq4, MACD_Sample_M.ex4
\experts\scripts	MultiTester_Distribute.mq4 , MultiTester_Distribute.ex4 MultiTester_Delete.mq4 , MultiTester_Delete.ex4 MultiTester_Collect.mq4, MultiTester_Collect.ex4
\experts\include	MultiTester_v1.1.mqh
\experts\libraries	MultiTester_v1.0.ex4, MultiOptimize1.dll MultiOptimize2.dll , kill.exe
\experts\files\MultiTester	empty path where logs and reports will be written into
\tester\files\MultiTester	empty path where logs and reports will be written into

Next you need to run the guardian.exe for the copy protection. The library files are encrypted for copy protection and the guardian.exe will decrypt them when needed for the MT4 terminal. You need to register with name and E-mail address for guardian to run properly. Guardian will check the license when you boot your computer, so you will need internet connection at that time. However if disconnected after that you can also work without internet connection. See the README_GUARDIAN.txt for more information.

Now start the terminal of your MT4 master installation. Select the MultiTester_Control EA on any symbol and time frame in the tester window and open the property sheet:



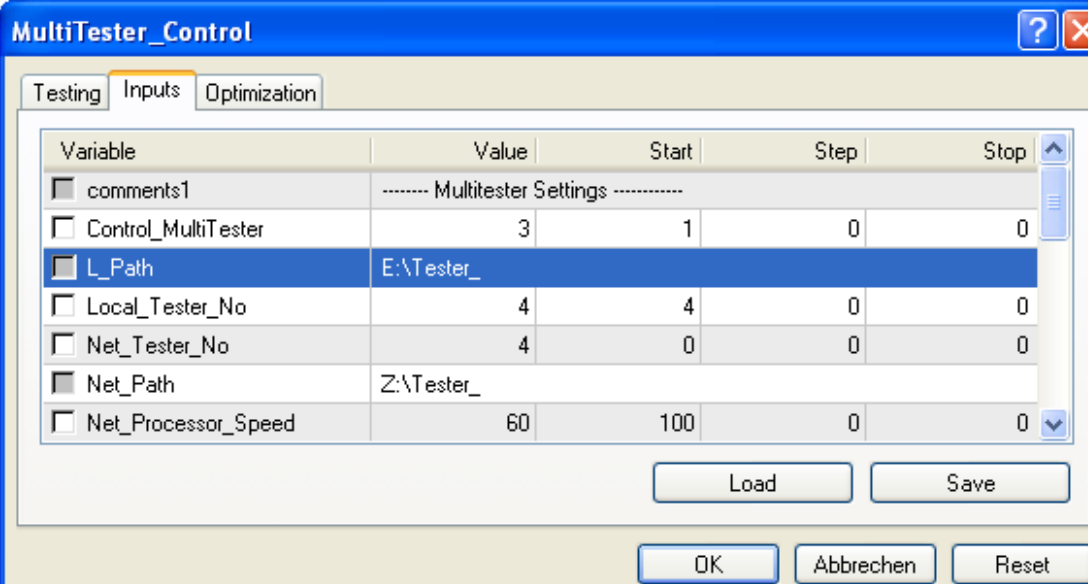
Variable	Value	Start	Step	Stop
☐ comments1	----- Multitester Settings -----			
☐ Control_MultiTester	1	1	0	0
☐ L_Path	Enter full path and base name for local tester instances			
☐ Local_Tester_No	0	4	0	0
☐ Net_Tester_No	0	0	0	0
☐ Net_Path	Enter full mapped path and base name for remote tester instances			
☐ Net_Processor_Speed	100	100	0	0

Buttons: Load, Save, OK, Abbrechen, Reset

Set the string parameter L_path to the path and name you want to have your local slave terminals installed. For example if you want to have your slave testers on the root of partition E: and if they should be named Tester_1 , Tester_2,... enter E:\Tester_ for L_Path.

Note: A path used must be a full root path and must not contain any blanks !

Enter the number of slave testers you want to use under Local_Tester_No. If you have a four core system you should enter 4.



Variable	Value	Start	Step	Stop
☐ comments1	----- Multitester Settings -----			
☐ Control_MultiTester	3	1	0	0
☒ L_Path	E:\Tester_			
☐ Local_Tester_No	4	4	0	0
☐ Net_Tester_No	4	0	0	0
☐ Net_Path	Z:\Tester_			
☐ Net_Processor_Speed	60	100	0	0

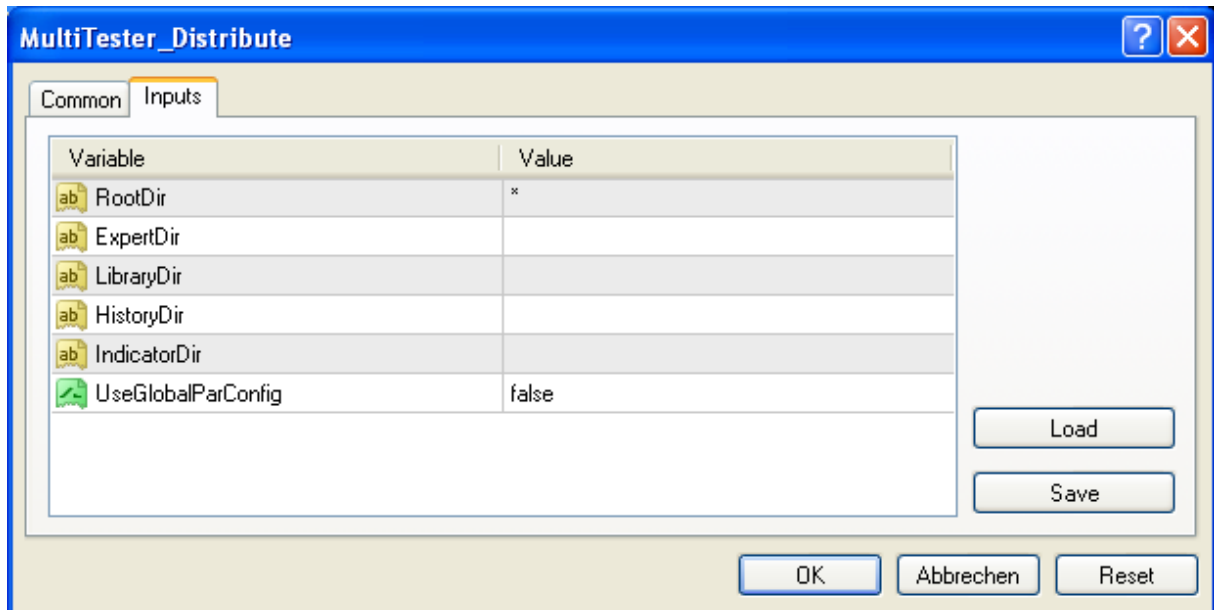
Buttons: Load, Save, OK, Abbrechen, Reset

In case you have a second computer in a local network you should map a drive or directory with the explorer for your local access. You must have write access on this drive.

Set the string parameter `Net_path` to the path of your mapped network drive and the name you want to have your remote slave terminals installed. For example if your mapped network drive is Z: and the tester should also be named `Tester_1` , `Tester_2`,... enter `Z:\Tester_` for `L_Path`.

Enter the number of remote slave testers you want to use under `Net_Tester_No`. If your remote computer has 8 cores you should enter 8. In the example above we have also entered 4.

Now close the property sheet and start the script `MultiTester_Distribute` on any chart. The property sheet with the parameters of this script will pop up.



Here you can enter in one of the `..Dir` lines the filename you want to distribute from this directory. Enter `*` in `RootDir` to distribute the full MT4 Master installation as slave installation.

UseGlobalParConfig: This parameter tells the script where to look for the settings of the MT4-MultiTester. If you use the `MutiTester_Control` EA set to false. If you use the full integrated user interface, set to true.

As we have configured the MT4-Multitester in the `MultiTester_Control` EA above, we have to set `UseGlobalParConfig` to false.

Start the script by click on `Ok`. The script will start some batch copy processes to distribute the files to all slave instances configured. Here the slave instances will be generated completely as full copy of the MT4 master installation. After all copy processes have been started, the script will terminate with a message, telling how many copies have been distributed. Note that despite of this message the copy process may last a while, depending on the amount of data (history) in your Mt4 master installation.

With our entries above at the end you should have four local slave instances located in `E:\Tester_1` , `E:\Tester_2`, `E:\Tester_3` and `E:\Tester_4` .

*Note: It's a good idea to clean up the master tester instance before distributing it to the slaves. Delete all *.txt files in \tester\history and all old log files. You could also remove all unnecessary history data in \history. You can always distribute history data later by use of*

the MultiTester_Distribute script. It is also recommended to minimize your terminal appearance before distribution in order to show the tester only in smallest possible size.

Run a multithreaded optimization

To run a multithreaded optimization select the EA you want to optimize in your tester and open the EA property sheet. Set the parameters you want to optimize as usual. Don't forget the checkbox to indicate which parameter shall be altered.

Now close the property sheet. On close the MT4 terminal saves your setting automatically. Therefore there is no need to store your setting manually.

Now select the MultiTester_Control EA, open the property sheet to complete the MultiTester configuration:

The screenshot shows the 'MultiTester_Control' dialog box with the 'Optimization' tab selected. It contains a table with columns: Variable, Value, Start, Step, and Stop. Below the table are 'Load' and 'Save' buttons. At the bottom are 'OK', 'Abbrechen', and 'Reset' buttons.

Variable	Value	Start	Step	Stop
☐ Net_Processor_Speed	60	100	0	0
☒ Server	Tadawulfx-Demo			
☒ Force_Spread_equal	true			
☒ Shutdown	true			
☐ Account_No	999568479	0	0	0
☒ Account_Password				
☐ TestModel	0	0	0	0

Server: Enter the name of the subdirectory of /history where your terminal takes the history data from.

Force_Spread_equal: Set to true if you want to start the slave testers unconnected, all with the same spread setting as your master tester was set to at start of the test. In this case the terminals will be show an enumerated account name. In This mode you could also use some of the available spread change tools in order to control the spread with for the optimization.

If you want to start your slave tester terminals connected you should set Force_Spread_equal to false. In this case you should enter your Account_No and Account password in the next two lines. Note that the salve testers will not start all at exactly the same time. Therefore it could happen, that they run with different spread, if the broker has just altered the spread between the actual starting points.

Net_Processor_Speed :This parameter allows a load balancing in case your local computer and the remote one on the network have different performance. For example if your remote computer has 60% of the performance of your local one put in 60 here. By this the remote

one will get accordingly less optimization runs. This will ensure that all slave testers are finished after about the same computing time.

Shutdown: Set to true to close all your slave tester automatically at the end of the test. If set to false the will stay and you have to close them manually or with the script MultiTester_KillSlaves.

Note: As the slave tester instances are initially identical copies of your master tester installation they will pop up at the same location one over the other. In order to monitor the progress you can distribute them nicely and eventually remove all unnecessary displays during the first optimization run.

Now the final settings needs to be done in the MultiTester_Control EA property sheet:

Variable	Value	Start	Step	Stop
☐ TestModel	0	0	0	0
☐ comments2	----- EA Settings -----			
☐ EA_Name	MACD Sample			
☐ From_Date	2011.12.01			
☐ To_Date	2012.02.17			
☐ Pair	GBPUSD			
☐ TimeFrame	15	1	0	0

Buttons: Load, Save, OK, Abbrechen, Reset

Enter the name of the EA you want to test for EA_Name, the start and end date of your test under From_Date and To_Date, the currency pair you want to test under Pair and the time frame in TimeFrame according to the MT4 allowed values (1= 1 minute ,5 = 5 minutes,...). The tester model is entered under TestModel (0 for "every tick",1 for "control points" or 2 for "open prices")

The main control of the MT4-MultiTester is the parameter

Control_MultiTester:

- 0 - MT4-MultiTester inactive
- 1 - Standard operation with use of tester cache
- 2 - Force start of slave testers even with no change in the setting
- 3 - Delete all slave tester caches before start to force a new calculation

Now simply start the MultiTester_Control EA for a single run (no optimization). This can be done on any instrument, timeframe or period, just make sure the history data is there, so that the MultiTester_control EA will run. However to avoid long waiting time for the tester start it is convenient to use a short test duration setting and a high timeframe for the MultiTester_Control EA.

The control EA will start the configured slave testers, each working on a share of the defined optimization task.

Get the results

To collect the results in a merged report the script MultiTester_Collect needs to be started on any chart of your MT4 master terminal. This script will wait for the slave tester to finish their work and collect the results to one merged csv document.

Finally this merged document will be opened with the application configured in the MultiTester_Collect script parameters. You can also configure this script to run indefinitely, so you will not need to start it again for another Multi Optimization.

The parameter settings for the MultiTester_Collect Script are shown above. Here the detailed explanation of these parameters:

NameResultFile: First part of the name of the result file. Full name in this case will be Merged_Report_hh_mm.csv , where hh is the hour and mm the minutes of the time the report was created. The file can be found in the \experts\files\MultiTester directory of your master MT4 instance.

SortForRow: This parameter controls how the optimization results are sorted in the merged report. Entry of 0 means the sorting set in your EA's property sheet, where you have set up the optimization, is used (first tab of the property sheets). Any other number means sorting for the corresponding row of the report.

Application: This is the name of the application with which the merged result file will be shown at the end.

EndlessLoop: This controls whether the script runs endless and keeps waiting always for the next optimization results. If set to false the script stops after one complete merged report has been collected.

UseGlobalParConfig: This tells the script where to look for the settings of the MT4-MultiTester. If you use the MultiTester_Control EA set to false. If you use the full integrated user interface, set to true.

UseOldResults: Whenever a slave tester completes a test the script removes the tester report files from the slave tester instance and puts it to \experts\files\MultiTester\LocalReports for any local slave tester and to \experts\files\MultiTester\RemoteReports for any remote slave tester. Then the merged report is generated. If you want to generate the last report again set UseOldResults to true. Note that these old individual slave tester reports are overwritten with any new optimization.

Set up remote slave tester

To use the computing power of a remote computer in a network you need a shared directory with this computer and write access on this directory.

Then you should enter the mapped path of this directory in the MultiTester_Control EA property sheet as described above. Enter the number of slave tester instances and generate them with the script MultiTester_Distribute

Now you should move over to the remote computer or log in with any remote control tool. On your remote computer open the file "MultiTester_Remote.bat" for editing and enter the full path of the directory where all your slave tester instances are in in the second line.

For example if your slave tester instances are in C:\MT4\tester_1, C:\MT4\tester_2,... the first two lines of "MultiTester_Remote.bat" should look like this:

```
@ECHO off
set p=C:\MT4\
```

Alternatively you could also provide this root path as parameter or as requested user input on start of the batch file.

Now start this batch file. Depending on the security setting of your computer you may not be able to start it within the shared drive. In this case copy it to any convenient location and start it there. This batch file waits for command from your master tester and start for example the slave tester on your remote pc. This rather simple solution was chosen as it works with all newer windows operating system without the need to be an expert in network administration. The required setup of a shared network folder is very well documented in the available windows documentations and notes.

Now you can start your optimization as described above with the MultiTester_Control EA. On start on your remote and on your local PC the configured number of slave testers will pop up.

Use the full tester integration

As alternative user interface the full integration into the MT4 tester is also provided. This has the advantage that you have almost the identical workflow when doing optimizations with the MT4_MultiTester. Just load your EA into the tester, set all parameters as usual and start a backtest as usual. If configured properly your EA will start all your slave testers through the MT4-MultiTester and idle through empty waiting loops then until you stop it.

To use this user interface you need to have the source code of your EA. In this source code you should enter the following three lines:

At the beginning, before the init() function enter:

```
#include <MultiTester_v1.1.mqh>    // Import MultiOptimize library for
parallel Optimization
```

As first statement of your init() function enter:

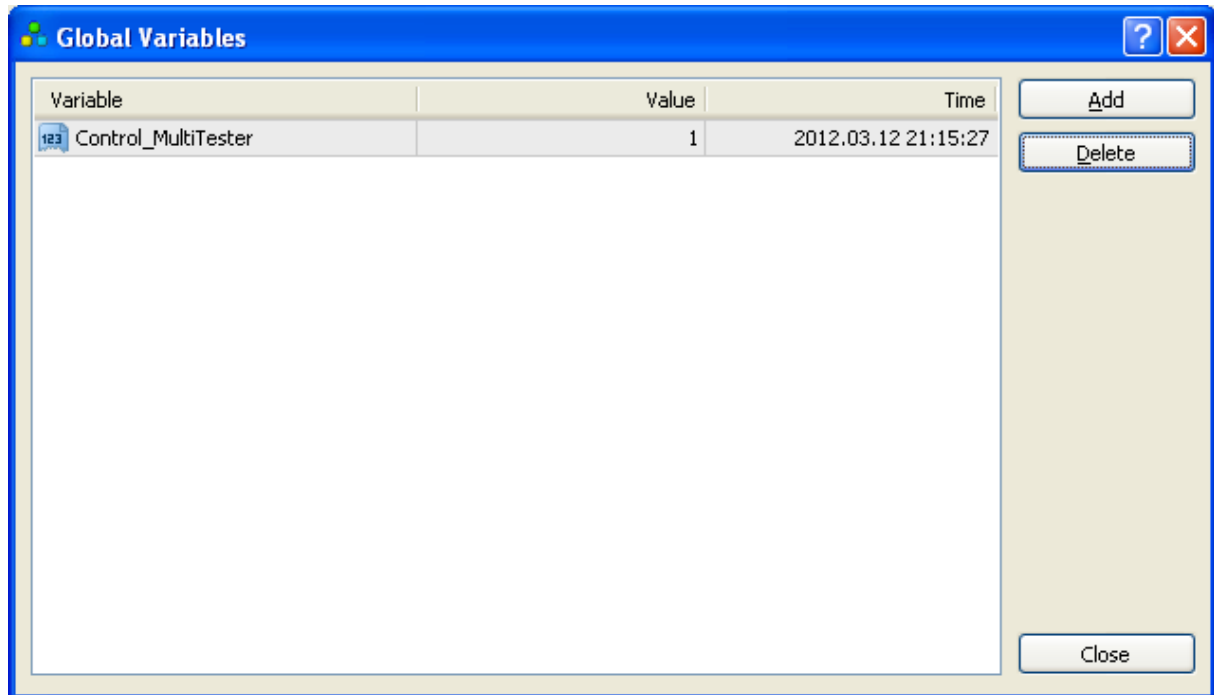
```
RunMultiTest = MultiTesterStart();    // call MultiTester depending if
global variable is set properly
```

and as first statement of your start() function enter:

```
if(RunMultiTest) return(0);            // No Work in master tester if
MultiTester works
```

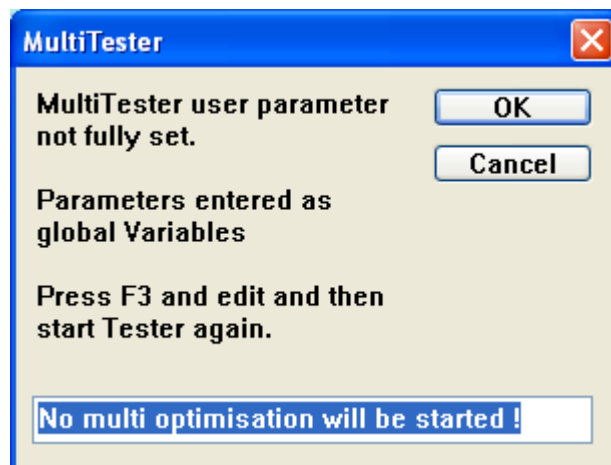
See the Source code of "Macd_Sample_M.mq4" where this change has been done for the standard Macd_Sample.mq4 delivered with every Metatrader.

Compile your EA , load it into the tester. Now press F3 to edit the global parameter and enter Control_MultiTester with value 0.

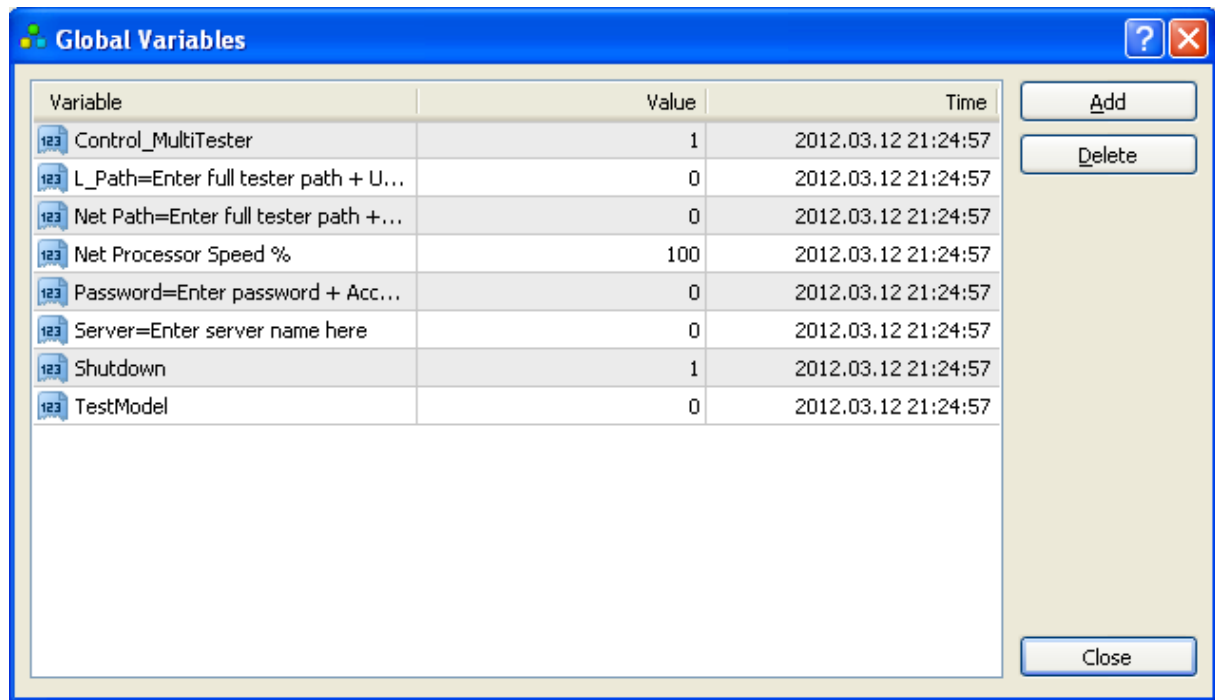


Note: If you do not enter this global parameter or if it's value is 0 the MT4-MultiTester is inactive. Your EA will behave like before with no action and no additional workload for the CPU beyond the single if statement in the start() function.

Now start an optimization of your EA. The following pop up will come up:

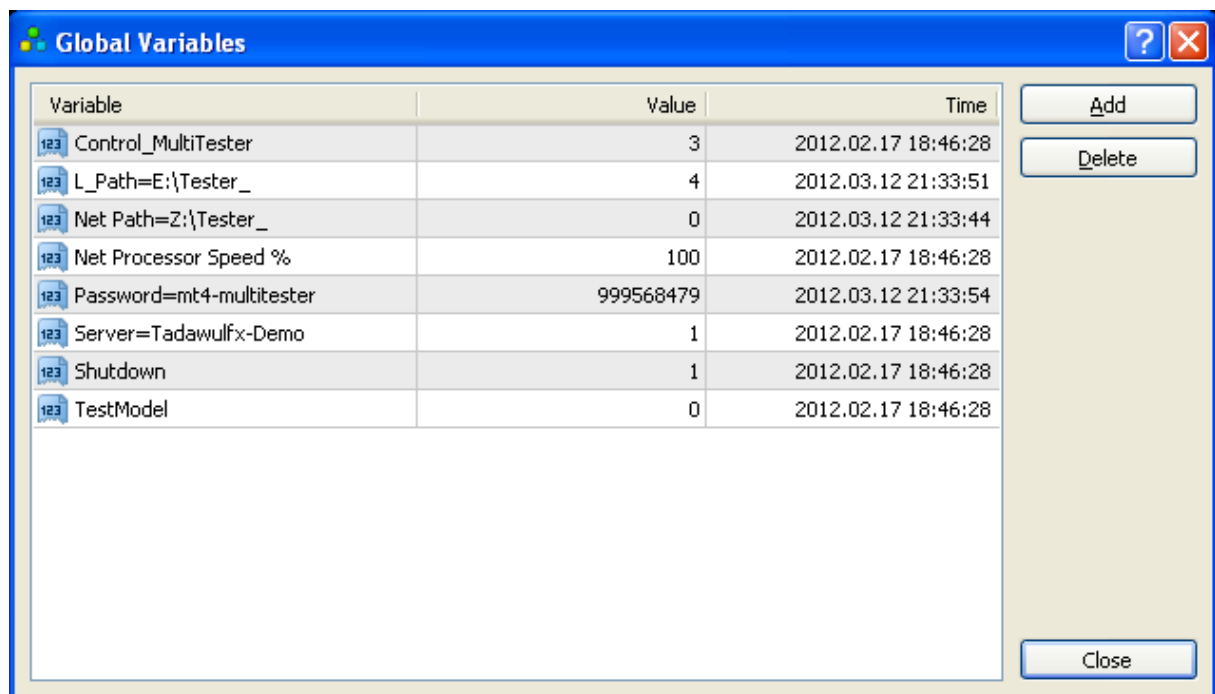


Click ok, stop the optimization and press F3 again to edit the global variables prepared now to control the MT4-MultiTester. You will get the following list:



The parameters are the same as in the MultiTester_Control EA property sheet as described above. Text parameters are entered at the left side behind the = sign. The number of local and remote slave instances to use are the value of the respective path.

With the same setting as used above for the MultiTester_Control EA the global parameters will look as follows:



The value 1 for the Server=... parameter represents a true for the **Force_Spread_equal** parameter, a value of 0 is equal to false. Same for the parameter Shutdown.

TestModel: If set to -1 the test model as set in your EA at time of optimization start is used . If set to 0 ,1 or 2 this setting in your EA will be ignored, This is for convenience to allow a

Testmodel Optimzation with another setting in your EA to speed up the start of the slave tester instances.

The global Parameter Control_MultiTester has the same function like in the MultiTester_Control EA with some additional options

Control_MultiTester:

- 0 - MT4-MultiTester inactive
- 1 - Standard operation with use of tester cache (start with optimize)
- 2 - Force start of slave testers even with no change in the setting (start with optimize)
- 3 - Delete all slave tester caches before start to force a new calculation (start with optimize)
- 4 - Standard operation with use of tester cache (start with single test)
- 5 - Force start of slave testers even with no change in the setting (start with single test)
- 6 - Delete all slave tester caches before start to force a new calculation (start with single test)

The options 4 to 6 have been added for debug purposes.

Note that due to the full integration into the MT4 Tester the software checks at every new run through the init() function whether the slave testers need to be started. With Control_MultiTester set to 1 this can be done reliably by checking for changes in the setting since the last start. with the settings 2 and 3 only the time passed since the last call can be used. This minimum waiting time is set to 20 seconds. Therefore a new salve tester run can only be started with 20 seconds delay after the last run.

The script MultiTester_Collect is again used to produce a merged report at the end, however as the configuration is now stored in the global variables this script need to be started with the parameter "UseGlobalParConfig" = true.

Tools

In case you want to stop an optimization for any reason, the script MultiTester_KillSlaves does the job, so you don't need to manually close all your slaves. Again this script can be started on any instrument and timeframe.

To remove single files or all files of your slave tester instances the script MultiTester_Delete is provided. Note that this script will leave empty directories, so you will need to delete them manually.

Note: The script MultiTester_Distribute is very powerful as it copies both, files and directories. This lead to a limitation, that it cannot overwrite a fully named already existing file. If you need to do this, for example in order to update the file EURUSD1.hst with the newest version for all slaves you should enter EURUSD1.h to do the job with the script.*