Trading Logic Engine
(TLE)

for

T101 Basket Trading

john_wustrack@hotmail.com

Table of Contents

Overview

While I was looking at the various threads and posts regarding trading the T101 methodology, one thing became apparent; there was no definitive trading methodology applied, there were no instances of if x crosses y then buy z for example.

In fact everyone seemed to trade the method (especially with the introduction of the T101 charts) in whatever form they preferred and on different time frames.

To create an automatic trading robot, that could handle this non-specific way of trading was simply not possible, nor was it feasible to create multiple versions of the robot for different trading methods and time frames.

Therefore I decided to create a new concept for the T101 Basket Trading robot, which I decided to call Trading Logic Engine (referred to hereafter as TLE).

The idea is simple, the basket handling part of trading (the opening, closing, trailing, hedging etc.) would be handled by the main robot.

All trade logic would be handled by separate indicators that had a uniformed entry & return structure.

This would allow the same robot to run several strategies, without the need for multiple versions of the robot.

The basic design

All TLE indicators have the same entry and return structure. All TLE indicators can do one or more of the following:-

- Specify that a basket should be immediately opened for buy / sell
- Specify that a basket should be set to be opened for buy / sell dependent on a certain price limit being breached (typical pending type orders, stop & limit)
- Send alerts to the user
- Perform different logic dependent on the read of the market
- Close baskets

To achieve this, the TLE indicator is ALWAYS passed the current Market Read (which is set by the user), the mode that the indicator should run in (look for openings, look for closing), and the basket type (buy / sell) if in closing mode.

The indicator will ALWAYS pass back to the Trading Robot, the action to take (including if no-action is to be taken). If there is an action to take that involves the opening of a pending type order then the price that the order should be opened at, will also be sent back.

Default TLE's provided

To help simplify the process I have provided two default TLE's, one requests orders immediately, the other requests that pending orders be set.

In both example TLE's, the logic is based on how the candles react with the Bollinger bands.

!TLE_Example_Immed

This will open buy & sell baskets dependent on the price reactions to the bollinger bands, it will also send alerts – the open logic will only work if the market read is set, if the market read is set to not applicable then it will assume a market read of range. If the market read is set to none, nothing will be returned.

It will only send closure requests for baskets when the market read is set to range. It will send warning alerts though when price reaches the opposite bollinger band, as long as the market read is not none.

!TLE_Example_Pending

This will send pending limit buy & sell baskets dependent on the price reactions to the bollinger bands, it will also send alerts – the open logic will only work if the market read is set, if the market read is set to not applicable then it will assume a market read of range. If the market read is set to none, nothing will be returned.

It will only send closure requests for baskets when the market read is set to range. It will send warning alerts though when price reaches the opposite bollinger band, as long as the market read is not none.

Creating your own TLE's

Using the examples provided it should be very simple for you to design and create your own TLE's (assuming of course you can program in MQL4).

There are a few guidelines though that should be kept in mind

- You cannot pass the custom TLE any more parameters, if you wish to have the TLE have user definable parameters without editing the code then something like Global Variables must be used.
- To make it easier for other people to recognize a TLE, use a naming convention of "!TLE"