

Expert Advisor based on the "New Trading Dimensions" by Bill Williams

[Alexey Klenov](#) | 11 March, 2011 | Views: 8542

Introduction

In this article I will discuss the development of a trading Expert Advisor, based on the book [New Trading Dimensions: How to Profit from Chaos in Stocks, Bonds, and Commodities](#) by B. Williams for the [MetaTrader 5](#) platform in the MQL5 language. The strategy itself is well known and its use is still controversial among traders.

It attracts newcomers with its ready study of signals that are "almost entirely" deprived of a subjective side, in contrast to, for example, the interpretation of the Elliott wave theory. But this is only at first glance, in some places the decisions will still have to be made by the trader. And this will be discussed further in the article.

The objectives of this article:

- To develop, using [OOP paradigm](#) (Object-oriented programming), a class of EA, which implements trade, based on the strategy of B. Williams. We'll call it [C_TS_BW](#);
- In the class [C_TS_BW](#) when possible, use ready-made codes from the [Standard Library](#).
- Write an EA, which uses the class [C_TS_BW](#);
- Test the developed EA in the [Strategy Tester](#) on several Forex and CFD instruments;
- Finally, to confirm or refute the aptitude of this strategy in the current market conditions.

1. Indicators

In the base of the trading system there are signals from 4 indicators:

1. [Alligator](#);
2. [Fractals](#);
3. [The Awesome Oscillator](#);
4. [Acceleration/Deceleration](#).

1.1. Alligator

The [Alligator](#) technical indicator is a combination of Balance Lines ([Moving Averages](#)), which use fractal geometry and nonlinear dynamics.

- **The blue line (the Alligator's Jaw)** - is the Balance Line for the time period that was used for building a chart (a 13-period smoothed moving average, shifted by 8 bars into the future);
- **The red line (the Alligator's Teeth)** - is the Balance Line for a significant time period, lower by an order (8-period smoothed moving average, shifted by 5 bars into the future);
- **The green line (Alligator's Lips)** - is the Balance Line for a significant time period, lower by another order (5-period smoothed moving average, shifted by 3 bars into the future).

The Lips, Teeth and Jaws lines of the Alligator illustrate the interaction of different time periods. Since the market trends can be identified only for 15-30 percent of the time, we must follow the trends, and not work on markets that fluctuate only within certain price periods.

When the Jaws, Teeth and Lips are closed or intertwined, the Alligator is going to sleep or is already sleeping. When it sleeps, its hunger grows - so the more it sleeps, the hungrier it will be when it wakes up. When it wakes up, the first thing it does is it opens its mouth and begins to yawn. Then it begins to hear the smell of food: meat of a bull or bear, and begins to hunt for it. When the Alligator satisfies its hunger, it begins to lose interest in food-price (the Balance Lines join together) - this is the time to fixate the profit.

1.2. Fractals

All of the markets are characterized by the fact that, for the majority of time, the prices do not fluctuate much, and only for a short time period (15-30 percent) trend changes can be seen. The most favorable periods for the extraction of profit are when the market prices change in accordance with a certain trend.

Fractals ([Fractals](#)) - is one of the 4 indicators of Bill Williams' trading system, which allows to detect the bottom or and the top. The technical definition of an upwards fractal is a series of at least five successive bars, in which there are two bars, before and after the highest maximum, that have lower maximums. The opposite configuration (series of five bars, in which before and after the lowest minimum there are two bars with higher minimums) is a downward fractal. On a chart the fractals have the values of High and Low, and are indicated by upwards or downwards arrows.

The signals of the technical indicator Fractals need to be filtered, using the technical indicator Alligator. In other words, we should not conclude a deal to buy if a fractal is located below the Alligator's teeth, and we should not conclude a deal to sell if a fractal is located above the Alligator's Teeth. After the fractal signal is formed and is in force, which is determined by its position beyond the Alligator's Jaws, it remains a signal until it is hit, or until the appearance of a more recent fractal signal.

1.3. The AO (Awesome Oscillator)

The Awesome Oscillator technical indicator by Bill Williams ([Awesome Oscillator, AO](#)) - is a 34-period simple moving average, built by the average points of the bars $(H + L)/2$, which is subtracted from the 5-period simple moving average, which is built by the central points of the bars $(H + L)/2$. It let's us know what is happening at the current moment with the market's driving force.

1.4. AO (Accelerator Oscillator)

Price is the last element that changes. Before the price changes, the driving force of the market changes, and before the driving force changes its direction, the acceleration of the driving force should slow down and reach zero. Then it begins to accelerate until the price begins to change direction.

The Acceleration/Deceleration technical Indicator ([Accelerator/Decelerator Oscillator, AC](#)) measures the acceleration and deceleration of the current driving force. This indicator will change direction before the change of the driving force takes place, and it, in turn, will change its direction before the change in price takes place. Understanding that the AC is an earlier warning signal, has obvious advantages.

2. Signals

The trading system, described in the "[New Trading Dimensions](#)" book by B. Williams, uses signals from five trading dimensions.

- The first dimension: overcoming the fractal beyond the Alligator's jaws;
- The second dimension: the signals from the AO indicator (Awesome Oscillator);
- The third dimension: the signals from the AC indicator (Accelerator Oscillator);
- The fourth dimension: trade in zones;
- The fifth dimension: balance line trade.

More about the signals from each dimension.

2.1. Description of trading by signals of the first dimension

The "A" fractal (buy signal) is false, because the price breaking occurs below the Jaw line of the Alligator. The "B" fractal is executed and we have an opened short position. With the appearance of "C" fractal, we close our short position, and already have a net long position.

With the breaking of "D" fractal, we again switch the position from buy to sell. Overcoming "E" fractal to buy, the market once again tells us that the position needs to be changes from sell to buy. With the execution of signals from "G" and "J" fractals, we add one more contract to the long position.

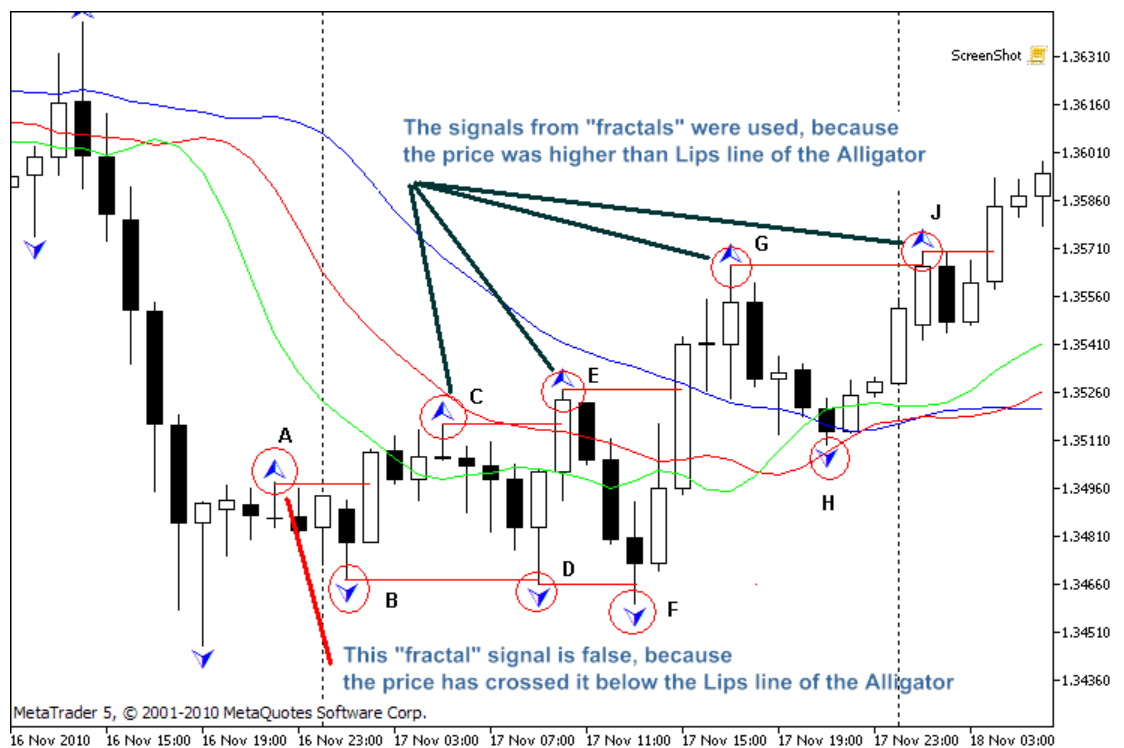


Figure 1. An example of trade in signals of the first dimension

2.2. Description of trade by signals of the second dimension

Two types of signals of the second dimension were implemented. This is the AO (Awesome Oscillator) indicator zero line crossover and the "Saucer" signal. Let's accept the [indexing of bars](#) as we do in MetaTrader 5, ie as [time series](#). The check of the signal will be done on the zeroth bar.

For the formation of a "Zero line crossover" pattern, there must be two bars of the indicator.

The buy signal is formed when the second bar of the indicator is below the zero line, and the first bar is above the zero line. The maximum price of the first bar will be the signal, and if it exceeds, the buy order will be sent to the server, using the current price. The sell signal is formed in the opposite case crossover.

For the formation of a "Saucer" pattern, there must be three consecutive bars, and the second one should be a red color and above the zero line, while the first one must be green and higher than the second. Signaling price will be the highest price of the first bar, which will be valid until it is overcome by the current price or a new buy signal is formed. The sell signals of the "saucer" pattern are formed similarly but on the opposite side of the indicator.

For the formation of a "Saucer" pattern, there must be three consecutive bars, and the second one should be a red color and above the zero line, while the first one must be green and higher than the second. Signaling price will be the highest price of the first bar, which will be valid until it is overcome by the current price or a new buy signal is formed. The sell signals of "saucer" pattern are formed similarly but on the opposite side of the indicator.

The "Twin Peaks" and "Twin bottoms" signals are not considered in this article and will not be programmed in the EA because of the fact that their formation almost always occurs on the opposite side of the Alligator: the twin bottoms is below the jaws line, and the twin peaks is above this line. Trading on such signals contradicts the system is that in order not to feed the Alligator, we can not buy below the jaws, and we can not sell above the Alligator's jaws.

For a more detailed description of the signals I encouraged you to see the original source.

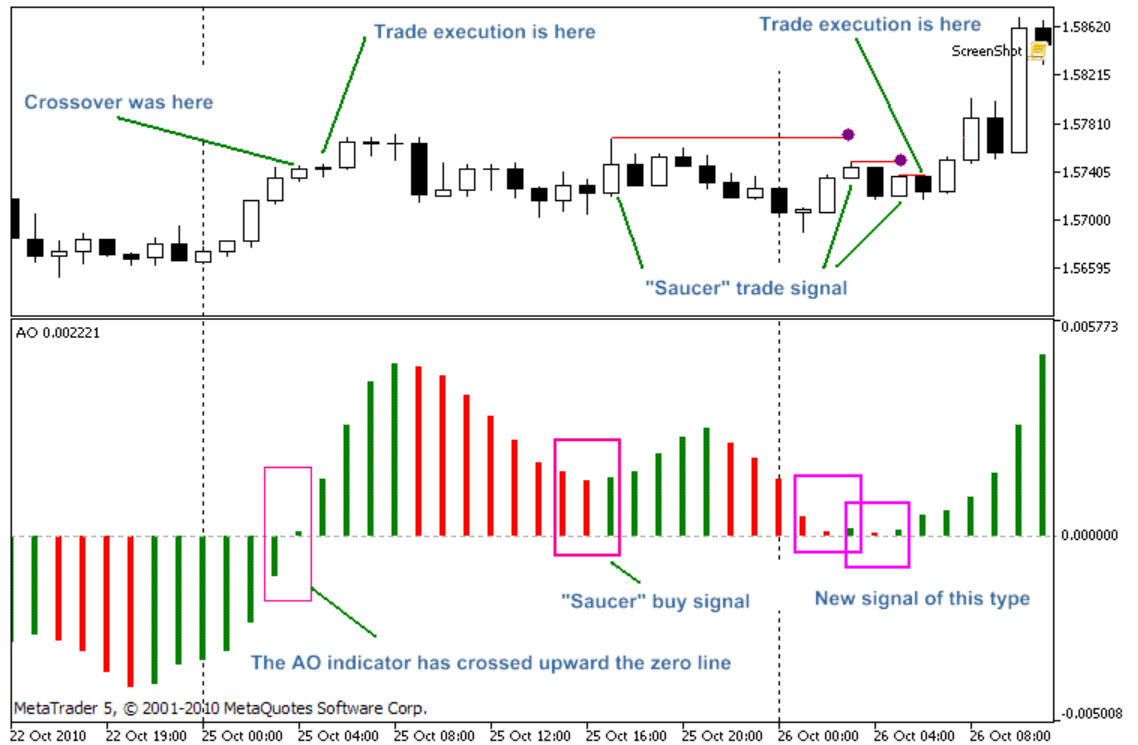


Figure 2. An example of trading signals of the second dimension

2.3. Description of trade signals of the third dimension

In the third dimension there are the "buy above the zero line" and "buy below the zero line" signals, formed by AC indicator (Accelerator Oscillator).

First we look at the pattern, specified in Figure 3. Checking is done for a zeroth bar. If the first bar is above the zero line ("A" case), which consists of two green and one red bar of a histogram. Meanwhile the position of the second and third bars, with respect to the zero line, doesn't matter.

If the first bar is below the zero line ("B" case), then we will need three green and one red bars for the formation of a buy signal. It also does not take into account the remaining bars, with respect to the zero line. The sell signals of the third dimension are inversely analogous.

For a more detailed description and examples, refer to the original source.

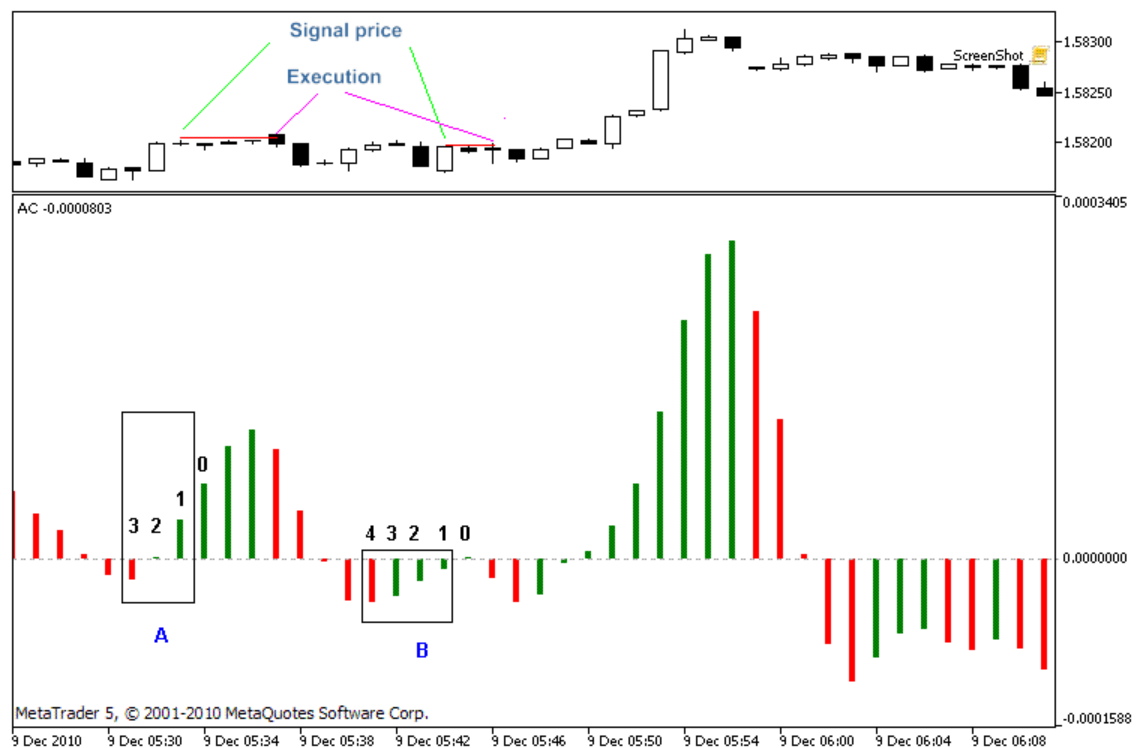


Figure 3. Example of trade of the third dimension signals

2.4. Description of trade signals of the fourth dimension ("Zone Trading")

The green zone is formed when two indicators (AO+AC) are green for a some bar.

The formation of a trading signal requires two consecutive green zone, and the closing price of the first bar must be higher than the closing price of the second bar.

Execution is implemented once a new bar is opened. In the original, the execution should be the order when the closing of a bar. There are situations when the last tick of the closing the bar can change the color of the AO or AC bars, and if this occurs, then it turns out that the buy according to the signal of the green zone was false. For these reasons, I used the opening of a new bar to buy.

Also, in the fourth dimension, there is a signal to trail a stop order for the long position.

This requires five consecutive green zones. The minimum value of the price of the first zone is used to set a Stop Loss. If the stop order does not triggered on the next bar (after it's closure), then we set the Stop Loss at the minimum price of the last completed bar (should be higher than the stop order) with the opening of a new one, without considering the zone color on the previous bar.

Also, we use a limitation on addition to the opened long position by the number of consecutive green zones: B. Williams recommends 6 or 8 zone. After this, we should wait for the appearance of a gray (this is when the colors of columns for AO and AC are different) or red zones, which will again allow to fill the position to buy from the green zone.

The signal to sell forms a "red zone" - a mirror image of the green zone. The color of the zone also affects the number of bars that form the signal from the "Balance Line Trade" (fifth dimension). For this line, B. Williams chose the "teeth" of the Alligator.

In this signal, I want to emphasize, that the OHLC prices of the zeroth bar participate in the formation of the signal.

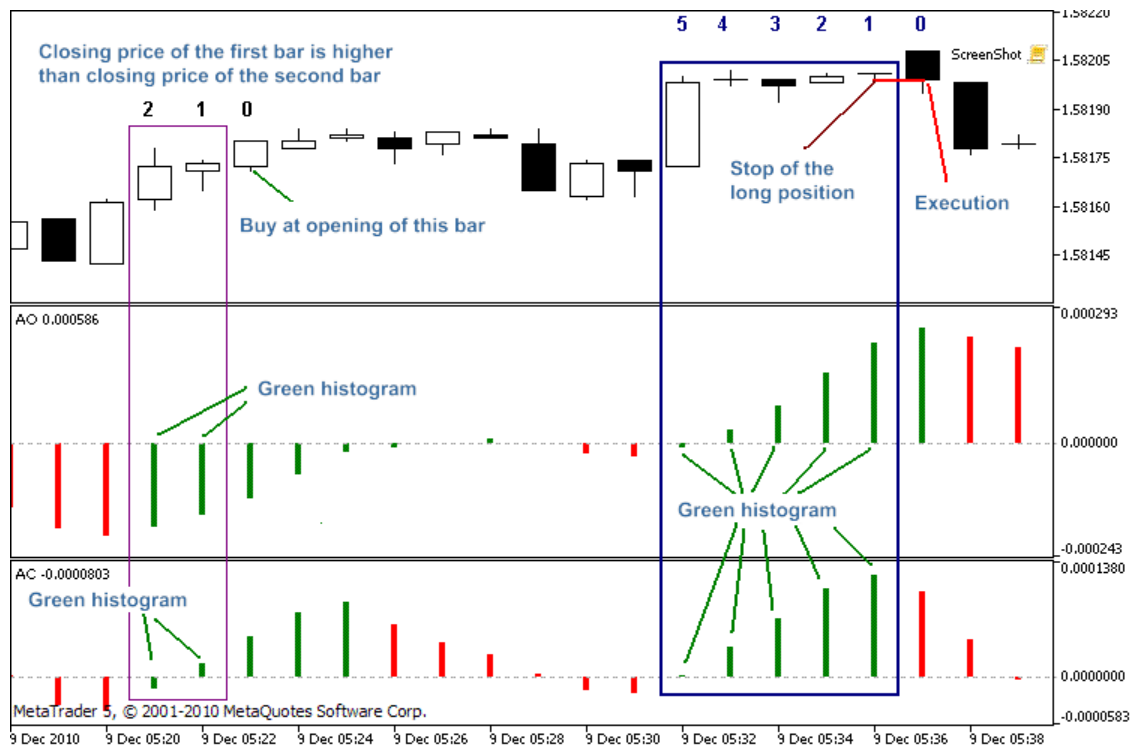


Figure 4. An example of trade of the fourth dimension signals

2.5. Description of trade signals of the fifth dimension

The "Buy above the balance line" pattern (if the zone green) is formed by two bars. If the opening price of a zero bar (also the highest price of the bar at this moment) is lower than the last highest price of the bar (can be found a few bars back), then the maximum price found, will be the price for opening a buy position in the green zone.

The red or gray zones require another maximum, higher then the price to enter in the green zone. As soon as we find it (usually less than 10 bars back and above the Alligator's teeth [I didn't find how many bars back must be looked for such a pattern in the author's text]), remember it as the price for entering in the red or gray zone, in the direction to buy.

To open position by the "buy above the balance line" signal, we check for the current price exceeding the price on the each new bar (for the green/red/grey zones).

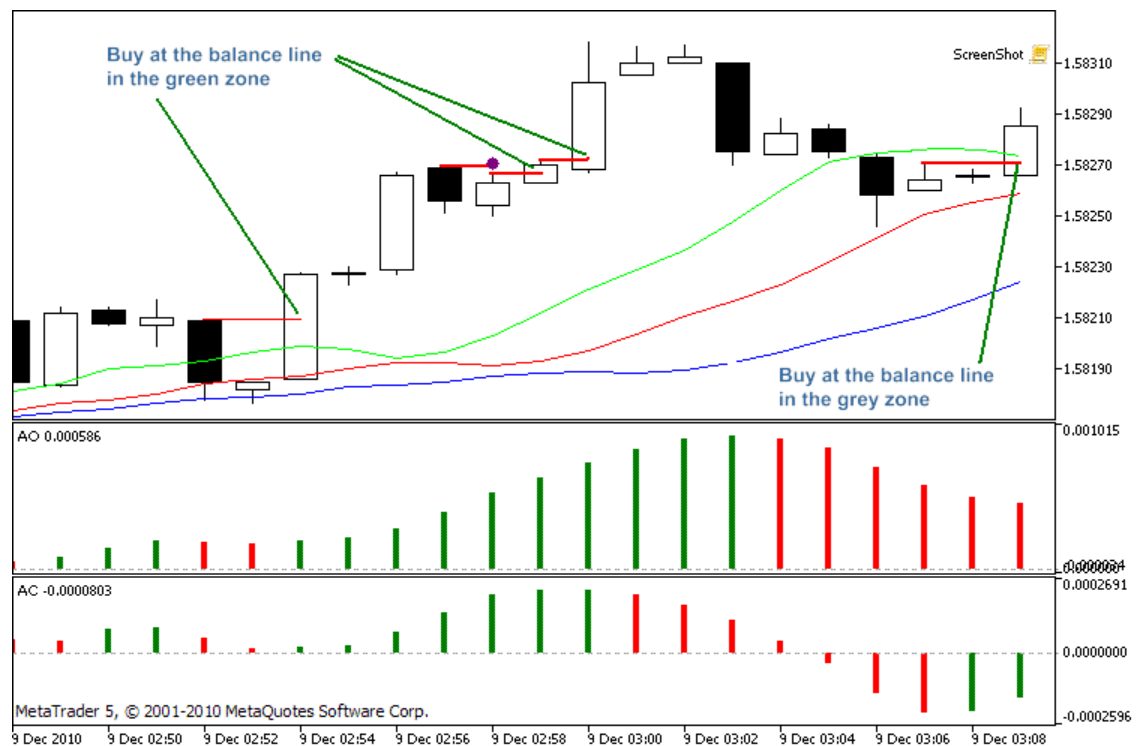


Figure 5. An example of trade of the fifth dimension signals

3. The C_TS_BW class

3.1. Class purpose:

- If possible, use the [Standard Library](#) classes;
- Large amounts of "similar data" should be stored in the structures;
- Customize settings;
- Obtain the required number of calculated data from the indicators for analysis;
- When a [new bar](#), check for the signals from the five trade dimensions;
- When a new tick, checks for the signal for opening a position;
- Calculate the lot, either fixed or "pyramiding" (you will learn about this algorithm in the method CalcLot of this class);
- Allow to change the lot of the EA directly for each type of a signal. Implement the custom need of money management algorithms;
- Trail a stop order of the opened position. If necessary, change the price of the Stop Loss, calculated by the user from the EA;
- Send order on the trade server and make a second request if error;
- Prohibit the duplication of trade signals (enter only one time per one signal);
- Minimize the number of public class methods;

3.2. The implementation of class C_TS_BW

- ...If possible, use classes and methods from the Standard library;

To perform this task, you first need to include the appropriate files from the [Standard Library](#).

This is implemented by the code below.

```
#include <Trade\Trade.mqh>
#include <Trade\SymbolInfo.mqh>
#include <Trade\PositionInfo.mqh>
#include <Trade\HistoryOrderInfo.mqh>
```

In the [private](#) section of the class, declare the objects for the organization of [trading requests](#), obtaining the information on the symbol and opened position, as well as accessing the history of orders.

```
CTrade      exp_trade; // trading methods from the Standard Library
CSymbolInfo s_info;    // methods of accessing the information on symbol
CPositionInfo pos_info; // methods of obtaining the information on position
CHistoryOrderInfo h_info; // methods of accessing the history of orders
```

- ... Large amounts of "similar data" are stored in the structures.

This class uses four structures, two of them are declared in the [private](#) section. These are the structures:

```
struct l_signals // structure of signals and time of the bar, on which they appeared
struct l_trade   // structure of the time of the last triggered signals
```

Also there are two structures in the [public](#) section:

```
struct s_input_params // structure of the tuning parameters
struct s_actual_action // structure of the appropriate trading order
```

The objects of these types of structures:

```
l_signals      last_signals;    // signals
l_trade        last_trade;      // processed signals
s_input_params inp_param;       // internal structure of the accepted settings
s_input_params inp_param_tmp;   // receiving structure of the settings (received through a link with the
s_actual_action actual_action;  // orders on the serve
```

More details about the code of the structures can be found in the attached header file.

- ... customize settings;

This task is implemented using the `Init` method from the `public` section. Parameters for calls from the EA:

```
string Symbol_for_trade      // symbol, on which the class will trade
ENUM_TIMEFRAMES Period_for_trade // the period of the chart symbol, on which the search for signals and i
s_input_params &inp_param_tmp // receiving structure of the class settings
```

In this method also occurs the initialization of the involved indicators and the organizing of the necessary buffers for receiving data from them.

- ... obtain the required number of calculated data from the indicators for analysis;

This can be done using the `CopyIndValue` method from the `private` section, the parameters

```
int type      // what we request (0- Alligator, 2 - AO, 3 - AC )
int countValue // amount of data
```

Depending on the parameter `type`, the receiving buffers are automatically substituted for the calculated data, organized as a time-series for class initialization.

- ... during the opening of a new bar ...

A search for signals will be performed only once after opening a new bar. To do this, we need to determine that moment. The `NewBar` method does this, it has no input parameters and in a situation when a received tick opens a new bar, it returns true, otherwise, it returns false.

- ... check for the signals from five trading dimensions;

For each of the trading dimension, I started a separate method. The signals of the first dimension are checked for by `FindSignal_1_dimension`.

Calling parameters:

```
int type      // direction of the fractal for the search (0- to buy, 1 to sell)
double &price_out[] // this array, after a successful search, will contain the fractal price
datetime &time_out[] // in this array, after a successful search, will contain the time of the bar on which
```

The analysis of the second dimension will be done by the method `FindSignal_2_dimension`. Calling parameters:

```
int type      // direction of the search (0-in the direction of to buy, 1-in the direction to sell)
int sub_type   // subtype of the search (0- signal of transferring the zero line, 1- sign "saucer")
double &price_out[] // this array, after a successful search, will contain the signal price
datetime &time_out[] // this array, after a successful search, will contain the time of the bar signal
```

The analysis of the third dimension is done using the method `FindSignal_3_dimension` with input/output parameters:

```
int type      // direction of the search (0-in the direction of to buy, 1-in the direction of to sell)
int sub_type   // subtype of the search (0- signal on two identical bars, 1- signal for "three ident
double &price_out[] // this array, after a successful search, will contain the signal price
datetime &time_out[] // this array, after a successful search, will contain the time of the bar signal
```

The processing of the fourth dimension is done by the method `FindSignal_4_dimension`, with input/output parameters:

```
int type      // direction of the search (0-in the direction to buy, 1-in the direction to sell)
int sub_type   // what we are looking for (0- signal from the zones, 1- trailing stop for five consecut
double &price_out[] // this array, after a successful search, will contain the signal price
datetime &time_out[] // this array, after a successful search, will contain the time of the signal bar
```

The fifth dimension is monitored by the `FindSignal_5_dimension`. Parameters of this method:

```
int type      // direction of the search (0-in the direction to buy, 1-in the direction to sell)
int sub_type   // subtype of the search (0- signal for two bars, 1- signal for three bars)
double &price_out[] // this array, after a successful search, will contain the signal price
datetime &time_out[] // this array, after a successful search, will contain the time of the bar signal
```

The entire checking is united by the `CheckSignal` method, it does not have input parameters and contains:

- Copying of data from the indicators;
- A checking for the closing of a position (if it is used) for some line of the Alligator;

- Reset of the signals of the previous bar;
- Search for active fractals in both directions;
- Depending on the availability of a position, search for the signals from the second to the fifth dimension, in the direction of an open position.

This method is declared in the [public](#) section, it needs to be called from the EA.

- ...with the arrival of a new tick, check for the possibility of signal triggering for position opening;

In this class, the method [CheckForTradeSignal](#) is implemented, which executes the search for the possibilities of entering a position by the current price.

Calling parameters:

```
int dimension      // number of the dimension (from the first to the fifth)
int type           // direction of the trade (0- buy, 1- sell)
int sub_type       // signal subtype from the trading dimension (described in the signal search)
double &price_out[] // signal price
datetime &time_out[] // time of the signal bar
```

If all of the regulation for the activation of a signal are followed, then we return [true](#), otherwise - [false](#).

The check of the capabilities of processing all of the signals is combined by the method [CheckActionOnTick](#), announced in the section [public](#), it needs to be called from the EA. There are no parameters for the call. Each successful signal is stored in the object [actual_action](#), they will later be processed in the method [TradeActualSignals](#).

- ...calculate the lot; either fixed or "pyramiding"

The [CalcLot](#) method is declared in the [public](#) section, and can be called from the EA. It is intended for the calculation of the lot and the further modification of the variable [Lot](#), declared in the [private](#) section of the class. Calling parameters:

```
bool external // for the installation of the external calculated lot, you need to direct to this variable true
               // then the lot will be set by the value ext_lot.
               // Inside this class (calculate the lot), the method is called with the parameter false.
double ext_lot // external size of the lot
int type // type of lot calculation
           // (0- starting lot (use the value that was transferred during the initialization of the class,
           // -1 turn over lot (the sum of the lot of the current position and the starting lot),
           // the values from 5 to 2 are used if the aggressive trading mode is turned on (meaning "pyramiding"
           // In this mode: 5 - starting, multiply by 5,
           // 4 - starting, multiply by 4,
           // 3 - starting, multiply by 3,
           // 2 - starting, multiply by 2,
           // 1 - the filling lot is equal to the starting lot
```

Now that we have considered the fixed lot, let's talk more about the "pyramiding" lot.

Let the starting lot equal to 0.1. For example, open by the starting lot for the signal from the fractal, outside the Alligator's jaws, the total position will be 0.1 lots. After which we begin to analyze the incoming signals from the second to five dimensions. As soon as the signal is triggered, fill into the open position by 0.5 lots (starting, multiplied by 5), and obtain the total volume of the position as 0.6 lots. During the next signal in the direction of the open open position, fill in by 0.4 lots, and the total position will be equal to 1.0 lots.

The next signal in the direction of the position will give us an additional 0.3 lots, and its volume will be 1.3 lots. The fifth fill-in will be done for 0.2 lots, and the total volume of the position will become 1.5 lots. The following fill-ins in the direction of the position will occur only by 0.1 lots.

This algorithm of money management (MM) was described by B. Williams in the [Trading Chaos](#). The possibility of installation of a user lot allows us to implement practically any capital management.

For a correct determination of the sequence of orders in a deal, I use various Magic numbers.

Number	Description of the order
999	Turn over order
1000	Starting order
1001	Filling order (starting X 1)
1002	Filling order (starting X 2)
1003	Filling order (starting X 3)
1004	Filling order (starting X 4)
1005	Filling order (starting X 5)

The class calculated the needed number (using the [CalcMagic](#) method), which later will be identifying the trading operation on the server before sending of a new order.

- ...Allow to change the lot from the EA, directly for each type of signal.

If desired, you can implement in the EA a query of the structure [actual_action](#), in which the trading signals are stores in the form of bool variables. Each type and direction is corresponded to by a variable. If we have a value of true, then for this tick, the class will try to produce a trading operation by the specified trading signal, after which the lot for this signal can be changed. For each tick of the price change, the class sends only one order for the open position or a filling.

The setting of a lot is possible in the EA after the call of the [CheckActionOnTick](#) method and before the call of the

[TradeActualSignals](#). Since on the current tick there can be a number of signals, awaiting execution, only one will be selected.

The sequence of signal execution:

- Closing the position;
- Execution of one of the signals from the fractal (opening, refilling, overturn of position);
- The "Saucer" signal;
- The "Zero line crossover" signal;
- AC "two equally-colored bars";
- AC "three equally-colored bars";
- Trade in zones;
- The Balance line (2 bar signal);
- The Balance line (3 bar signal);

It is therefore necessary to consider this sequence during the installation of the "user" lot size.

- ... Trailing stop of the opened position. If necessary, change the price of the Stop Loss, calculated by the user from the EA.

In the described class, there is maintenance of the stop price at the position, by the Trailing Stop method, meaning the pulling up of the Stop Loss, only in the direction of increasing the profits at the position for the triggering at this price. There are five models of maintenance:

- By the Alligator's Lips line;
- By the Alligator's Teeth line;
- By the Alligator's Jaws line;
- By five consecutive zones of the same color (green zones to buy, red to sell);
- External setting of stop price.

The fifth option of the installation of a Stop Loss is implemented by the method [SetStopLoss](#), with a single parameter [double &stoploss](#). The call must be made from the EA and before the executing method [TrailingStop](#), which checks the prices for modification of the position, and sends the request to the server. After a successful execution of this procedure, the value of the internal variable [StopLoss](#) is reset to -1.

- ...Control the sending of the order of the opening deal to the server, and if it is unsuccessful, make a second request.

For the sending of a trading order for opening, closing or turning over a position, the method [boolSendOrder](#) is used. The calling parameters:

```
ENUM_ORDER_TYPE type // direction of the trading operation
double &price_out[] // pointer to the signal price, after a successful execution of the operation, set in -
datetime &time_out[] // the pointer to the signal time, after a successful execution of the operation, set i
string comment // the commentary for the order
```

The testing of all of the trading signals is combined by the method [TradeActualSignals](#). The structure `actual_action` stores trading orders. After the successfully sending to the server the order ([SendOrder](#) returns true), reset the trading signal in the structure `actual_action`. The trading signal will be active until we get a positive response on the send.

There is also a function in the structure `actual_action` `init`, without parameters, which resets all of the current signals. This procedure is used when opening a new position or an overturn in the existing position occurs.

- ...Prohibit the duplication of trade signals (enter only once for each signal);

The `last_trade` structure stores the time of the last trading signal for each type and direction. Before you set a trading order into the `actual_action` structure, check to see whether this signal is already trading in the `last_trade` structure, and if so, ignore it. Thus, this provides the implementation of control of the "disposable" execution of a trading situation.

- Minimize the number of public class methods.

Here is a list of class methods that are available by calling from the EA:

```
void C_TS_BW(); // Constructor
bool Init(string Symbol_for_trade, // Initialization of the class
          ENUM_TIMEFRAMES Period_for_trade,
          s_input_params &inp_param_tmp);
bool NewBar(); // Check for a new bar on the current symbol\time-fr
void CheckSignal(); // Search for signals
void CheckActionOnTick(); // Collecting the desired actions on the current tick
void TrailingStop(); // Trailing Stop
void TradeActualSignals(); // Trading by the current signals
void SetStopLoss(double &stoploss); // Set Stop Loss
void CalcLot(bool external, double ext_lot, int type); // Calculation of the lot
```

Also available are the structures:

```
actual_action // Current trading orders for execution
inp_param_tmp; // Reception of the structure of settings
               // (receives the data by the link during the initialization of the class)
```

4. The implementation of the EA, using the class `C_TS_BW`

The first thing that must be done is to include the `h_TS_BW.mqh` file in the Expert Advisor.

```
#include <h_TS_BW.mqh>
```

After this, declare the object of `C_TS_BW` class. Let this be the `EA_TS_BW`.

You will also need the structure of the adjustable parameters such as `s_input_parameters`, for example, `input_parameters`.

Here is a description of the parameters, incorporated into this structure:

<code>input_parameters.alligator_jaw_period</code>	// The Alligator: period of the Jaws line
<code>input_parameters.alligator_jaw_shift</code>	// The Alligator: shift of the Jaws line
<code>input_parameters.alligator_teeth_period</code>	// Alligator: period of the Teeth line
<code>input_parameters.alligator_teeth_shift</code>	// Alligator: shift of the Teeth line
<code>input_parameters.alligator_lips_period</code>	// Alligator: period of the Lips line
<code>input_parameters.alligator_lips_shift</code>	// Alligator: shift of the Lips line
<code>input_parameters.add_1_dimension</code>	// Allow the addition by Fractals
<code>input_parameters.add_2_dimension_bludce</code>	// Allow the addition by the "Saucer" (A0) signal
<code>input_parameters.add_2_dimension_cross_zero</code>	// Allow the addition by the "Crossing the zero line" (A0) signal
<code>input_parameters.add_3_dimension_use_2_bars</code>	// Allow the addition by the "AC 2 bars" signal
<code>input_parameters.add_3_dimension_use_3_bars</code>	// Allow the addition by the "AC 3 bars" signal
<code>input_parameters.add_4_dimension_zone</code>	// Allow the addition by the red or the green zone
<code>input_parameters.add_5_dimension</code>	// Allow the addition by the Balance Line
<code>input_parameters.max_4_dimension_zone</code>	// The maximum amount of consecutive bars of zones of the same col
<code>input_parameters.trall_4_dimension</code>	// Allow a trail position using 5 consecutive bars of zones of th
<code>input_parameters.agress_trade_mm</code>	// Aggressive style of filling in an open position
<code>input_parameters.support_position</code>	// Type of trailing stop of the position
<code>input_parameters.lot</code>	// Trading lot

In the `OnInit()` section of the EA, you need to:

- Fill in all of the required values of the `input_parameters` structure with data that will then be passed to the class.
- Using the `Init` method of class, make its initialization. Example:

```
expert_TS_BW.Init(Symbol(),PERIOD_CURRENT,input_parameters)
```

In this case, the EA will work on the current symbol/period where it is installed.

Example of the `OnTick()` section of Expert Advisor:

```
// double S1[1];
if(expert_TS_BW.NewBar()) // new bar on the chart
{
    expert_TS_BW.CheckSignal(); // signal search
}
expert_TS_BW.CheckActionOnTick(); // check for the required actions on the current tick
//---***** the place of the beginning of external control of the lot, the stop
//--- example of setting the lot for trade by the signal from the zones

// if(expert_TS_BW.actual_action.zone_buy || expert_TS_BW.actual_action.zone_sell)
// {expert_TS_BW.CalcLot(true,0.11,0);}
//--- setting the stop by the parabolic

// CopyBuffer(h_parabolic,0,0,1,S1);
// if (S1[0]>0){expert_TS_BW.SetStopLoss(S1[0]);}
//---*****the place of the end of external control of the lot, the stop

expert_TS_BW.TrailingStop(); // pulling-up the stop (if necessary)
expert_TS_BW.TradeActualSignals(); // trading of current signals
```

In this case, the examples of external control of the stop price and the trading lot, for the execution of the class, are commented on.

5. Some tests on the history

The author of the book, based on which the EA was written, argues that this system is focused on stocks and commodity markets.

First, let's check the Expert Advisor on the CFD. Suppose the testing polygon will be part of the history of IBM. The system is aimed at the trending segments of quotes. I took the first available segment, on which a trend can be seen by the naked eye.

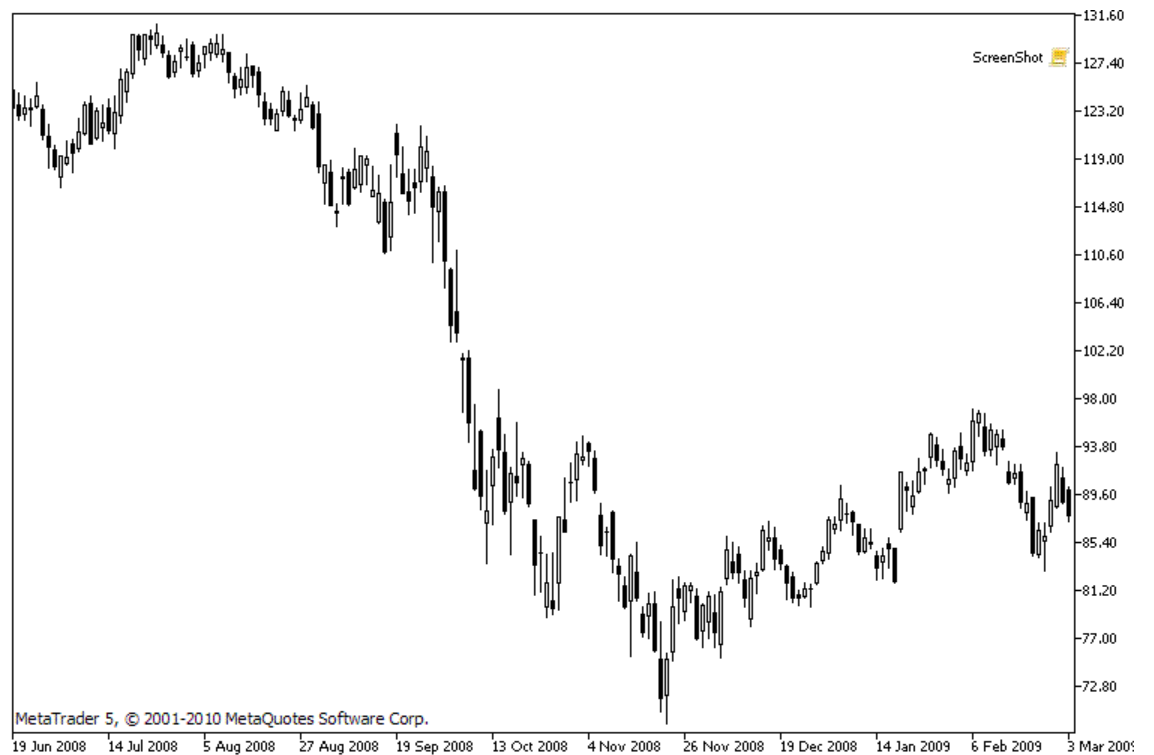


Figure 6. IBM chart

Run the written Expert Advisor on this segment and here is what we obtain as orders.



Figure 7. The trading signals of Bill Williams (IBM)

At first glance, there are many trades by the trend, which is good. Here is the chart, created by the Strategy Tester.

The trade was done on 0.1 lots, without closing by the Alligator lines, without a trailing stop using the five consecutive zones of one color and without aggressive trading (pyramiding).

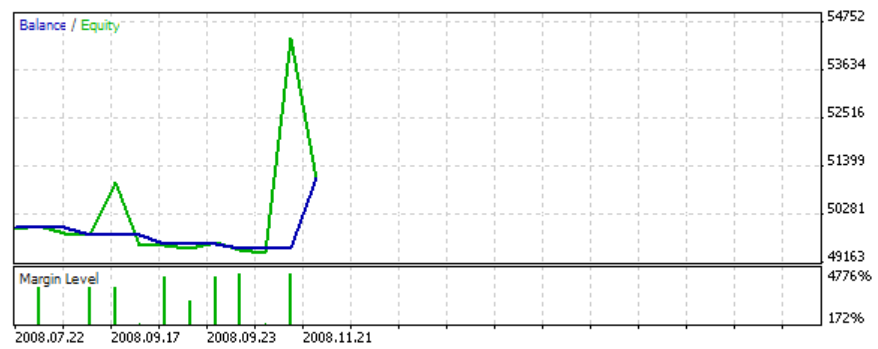


Figure 8. Test Results

In general, we obtained profit.

Now let's take a protracted trend with a lesser inclination.

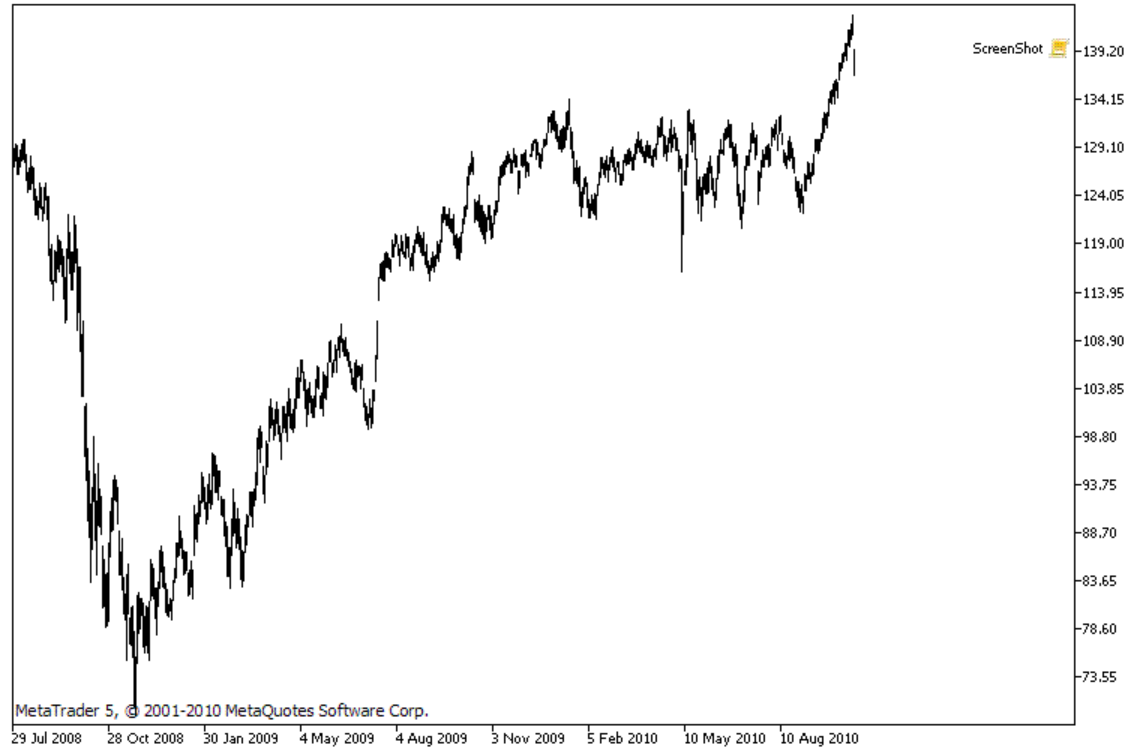


Figure 9. IBM chart (fragment 2)

Let this be the same symbol and period of 13 months (from 2008.12 to 2010.01)

Here is a chart from the Strategy Tester:

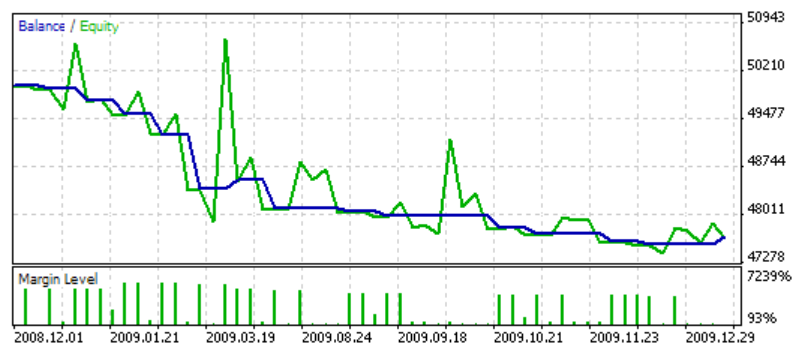


Figure 10. The results of testing the system on the history (fragment 2)

Let's just say, it's "not satisfactory" or "has not met our expectations."

Next I would like to check the work on the currency pairs.

Let's take the well-known EURUSD pair and the H1 period of the chart. The depth of the history is the year of 2010.

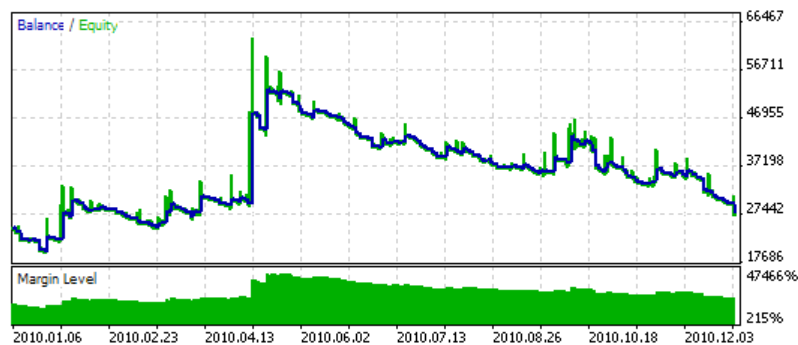


Figure 11. The results of testing the system on the history, EURUSD, H1, 2010

Let's try it on the daily bars of EURUSD for the same year (2010).

The report of the tester looks like this:

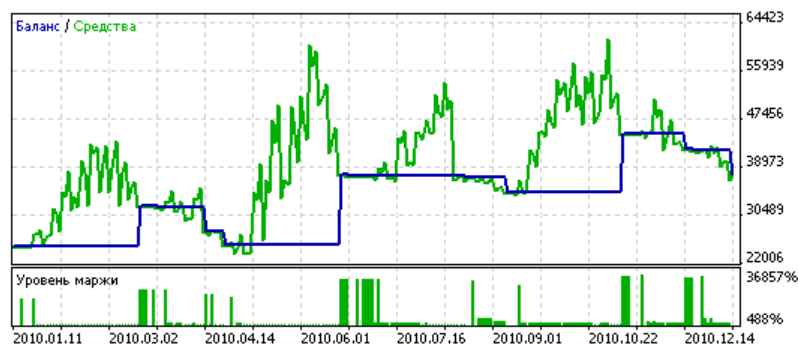


Figure 12. The results of testing the system on the history, EURUSD, D1, 2010

Conclusion

The aim of this article was to check the performance of one of the well-known trade strategies by Bill Williams, not only on the stock markets and commodity exchanges, but also on the Forex market. The system works "more or less" on the EURUSD daily charts of the past year, but brings no profit on smaller time-frames, without attempts of optimization.

The market entry signals of the system are quiet accurate (if we look at the day charts), but the exits are clearly delayed, since more than half of the profit is not fixed. This field is for the refining of the given system in terms of its optimization for smaller time-frames.

The complete code of the class can be found in the attached file.

Translated from Russian by MetaQuotes Software Corp.
Original article: <https://www.mql5.com/ru/articles/139>

Attached files | [Download ZIP](#)
[h ts_bw.mqh](#) (57.84 KB)
[tc_bw_new_version_2-02.mq5](#) (5.16 KB)

Warning: All rights to these materials are reserved by MQL5 Ltd. Copying or reprinting of these materials in whole or in part is prohibited.

Join us — download MetaTrader 5!

[Windows](#)

[iPhone/iPad](#)

[Mac OS](#)

[Android](#)

[Linux](#)

[MQL5 Strategy Language](#) | [Source Code Library](#) | [How to Write an Expert Advisor or an Indicator](#) | [Order Development of Expert Advisor](#)
[Download MetaTrader 5](#) | [MetaTrader 5 Trade Platform](#) | [Application Store](#) | [MQL5 Cloud Network](#)
[About](#) | [Website History](#) | [Terms and Conditions](#) | [Privacy Policy](#) | [Contacts](#)
Copyright 2000-2015, MQL5 Ltd.