

The Financial Hacker

A new view on algorithmic trading

Hacking a HFT system



Compared with machine learning or signal processing algorithms of conventional algo trading strategies, **High Frequency Trading** systems can be surprisingly simple. They need not attempt to predict future prices. They know the future prices already. Or rather, they know the prices that lie in the future for other, slower market participants. Recently we got some contracts for simulating HFT systems in order to determine their potential profit and maximum latency. This article is about testing HFT systems the hacker's way.

The HFT advantage is receiving price quotes earlier and getting orders filled faster than the majority of market participants. Its profit depends on the system's **latency**, the delay between price quote and subsequent order execution at the exchange. Latency is the most relevant factor of a HFT system. It can be optimized in two ways: by minimizing the distance to the exchange, and by maximizing the speed of the trading system. The former is more important than the latter.

The location

Ideally, a HFT server is located directly at the exchange. And indeed most exchanges are dear selling server space in their cellars – the closer to the main network hub, the dearer the space. Electrical signals in a shielded wire travel with about 0.7 .. 0.9 times the speed of light (300 km/ms). 1 m closer to the signal

source means a whopping 8 nanoseconds round-trip time advantage. How many trade opportunities disappear after 8 ns? I don't know, but people are willing to pay for any saved nanosecond.

Unfortunately (or fortunately, from a cost perspective), the HFT system that we'll examine here can not draw advantage from colocating at the exchange. For reasons that we'll see soon, it must trade and receive quotes from the NYSE and the CME simultaneously. There's a high speed cable and also a microwave radio link running between both cities. The theoretically ideal location of such a HFT system is **Warren, Ohio**, exactly halfway between New York City and Chicago. I don't know whether there's a high speed node in Warren, but if it is, the distance of 357 miles to both exchanges would be equivalent to about 4 ms round-trip latency.



Warren, Ohio – HFT Mecca of the world (image by Jack Pearce / Wikipedia Commons)

Without doubt, a server in this pleasant town would come at lower cost than a server in the cellar of the New York Stock Exchange. My today's free tip for getting rich: Buy some garage space in Warren, tap into the high speed New York-Chicago cable or radio link, and rent out server racks!

The software

When you've already invested money for the optimal location and connection of the HFT system, you'll also want an algo trading software that matches the required speed. Commercial trading platforms – even Zorro – are normally not fast enough. And you don't know what they do behind your back. HFT systems are therefore normally not based on a universal trading platform, but directly coded. Not in R or Python, but in a fast language, usually one of the following:

- **C or C++.** Good combination of high level and high speed. C is easy to read, but very effective and the fastest language of all – almost as fast as machine code.
- **Pentium Assembler.** Write your algorithm in machine instructions. Faster than even C when the algorithm mostly runs in loops that can be manually optimized. There are specialists for optimizing assembler code. Disadvantage: Any programmer has a hard time to understand assembler programs written by another programmer.
- **CUDA, HLSL, or GPU assembler.** Run your algorithm on the shader pipeline of the PC's video card. This makes sense when it is heavily based on vector and matrix operations.
- **VHDL.** If any software would be too slow and trade success really depends on nanoseconds, the ultimate solution is coding the system in hardware. In VHDL you can define arithmetic units, digital filters, and sequencers on a FPGA (Field Programmable Gate Array) chip with a clock rate up to several 100 MHz. The chip can be directly connected to the network interface.

With exception of VHDL, especially programmers of 3D computer games are normally familiar with those high-speed languages. But the standard language of HFT systems is plain C/C++. It is also used in this article.

The trading algorithm

Many HFT systems prey on traders by using fore-running methods. They catch your quote, buy the same asset some microseconds earlier, and sell it to you with a profit. Some exchanges prevent this in the interest of fair play, while other exchanges encourage this in the interest of earning more fees. For this article we won't use fore-running methods, but simply exploit [arbitrage](#) between ES and SPY. We're assuming that our server is located in Warren, Ohio, and has a high speed connection to Chicago and to New York.

ES is a Chicago traded S&P500 future, exposed to supply and demand. SPY is a New York traded ETF, issued by State Street in Boston and following the S&P500 index (minus State Street's fees). 1 ES Point is equivalent to 10 SPY cents, so the ES price is about ten times the SPY price. Since both assets are based on the same index, we can expect that their prices are highly correlated. There have been some publications (1) that claim this correlation will break down at low time frames, causing one asset trailing the other. We'll check if this is true. Any shortlived ES-SPY price difference that exceeds the bid-ask spreads and trading costs constitutes an arbitrage opportunity. Our arbitrage algorithm would work this way:

- Determine the SPY-ES difference.
- Determine its deviation from the mean.
- If the deviation exceeds the ask-bid spreads plus a threshold, open positions in ES and SPY in opposite directions.
- If the deviation reverses its sign and exceeds the ask-bid spreads plus a smaller threshold, close the positions.

This is the barebone HFT algorithm in C. If you've never seen HFT code before, it might look a bit strange:

```
#define THRESHOLD 0.4 // Entry/Exit threshold

// HFT arbitrage algorithm
// returns 0 for closing all positions
// returns 1 for opening ES long, SPY short
// returns 2 for opening ES short, SPY long
// returns -1 otherwise
int tradeHFT(double AskSPY, double BidSPY, double AskES, double BidES)
{
    double SpreadSPY = AskSPY-BidSPY, SpreadES = AskES-BidES;
    double Arbitrage = 0.5*(AskSPY+BidSPY-AskES-BidES);

    static double ArbMean = Arbitrage;
    ArbMean = 0.999*ArbMean + 0.001*Arbitrage;
    static double Deviation = 0;
    Deviation = 0.75*Deviation + 0.25*(Arbitrage - ArbMean);

    static int Position = 0;
    if(Position == 0) {
        if(Deviation > SpreadSPY+THRESHOLD)
            return Position = 1;
        if(-Deviation > SpreadES+THRESHOLD)
            return Position = 2;
    } else {
        if(Position == 1 && -Deviation > SpreadES+THRESHOLD/2)
            return Position = 0;
        if(Position == 2 && Deviation > SpreadSPY+THRESHOLD/2)
            return Position = 0;
    }
}
```

```
}  
    return -1;  
}
```

The **tradeHFT** function is called from some framework – not shown here – that gets the price quotes and sends the trade orders. Parameters are the current best ask and bid prices of ES and SPY from the top of the order book (we assume here that the SPY price is multiplied with 10 so that both assets are on the same scale). The function returns a code that tells the framework whether to open or close opposite positions or to do nothing. The **Arbitrage** variable is the mid price difference of SPY and ES. Its mean (**ArbMean**) is filtered by a slow EMA, and the **Deviation** from the mean is also slightly filtered by a fast EMA to prevent reactions on outlier quotes. The **Position** variable constitutes a tiny state machine with the 3 states long, short, and nothing. The entry/exit **Threshold** is here set to 40 cents, equivalent to a bad SPY spread multiplied with 10. This is the only adjustable parameter of the system. If we wanted to really trade it, we had to optimize the threshold using several months' ES and SPY data.

This minimalistic system would be quite easy to convert to Pentium assembler or even to a FPGA chip. But it is not really necessary: Even compiled with Zorro's lite-C compiler, the **tradeHFT** function executes in just about 750 nanoseconds. A strongly optimizing C compiler, like Microsoft VC++, gets the execution time down to 650 nanoseconds. Since the time span between two ES quotes is normally 10 microseconds or more, the speed of C is largely sufficient.

Our HFT experiment has to answer two questions. First, are those price differences big enough for arbitrage profit? And second, at which maximum latency will the system still work?

The data

For backtesting a HFT system, normal price data that you can freely download from brokers won't do. You have to buy high resolution order book or BBO data (**Best Bid and Offer**) that includes the exchange time stamps. Without knowing the exact time when the price quote was received at the exchange, it is not possible to determine the maximum latency.

Some companies are recording all quotes from all exchanges and are selling this data. Any one has its specific data format, so the first challenge is to convert this to lean data files that we then evaluate with our simulation software. We're using this very simple target format for price quotes:

```
typedef struct T1    // single tick
{
    double time; // time stamp, OLE DATE format
    float fVal;  // positive = ask price, negative = bid price
} T1;
```

The company watching the Chicago Mercantile Exchange delivers its data in a specific CSV format with many additional fields, of which most we don't need here (f.i. the trade volume or the quote arrival time). Every day's quotes are stored in one CSV file. This is the Zorro script for pulling the ES December 2016 contract out of it and converting it to a dataset of T1 price quotes:

```
////////////////////////////////////
// Convert price history from Nanotick BBO to .t1
////////////////////////////////////

#define STARTDAY 20161004
#define ENDDAY   20161014

string InName = "History\\CME.%08d-%08d.E.BBO-C.310.ES.csv"; // name of a day file
string OutName = "History\\ES_201610.t1";
string Code = "ESZ"; // December contract symbol

string Format = "2,, %Y%m%d, %H:%M:%S,, ,s,, ,s,i,, "; // Nanotick csv format
void main()
{
    int N, Row, Record, Records;
    for(N = STARTDAY; N <= ENDDAY; N++)
    {
        string FileName = strf(InName, N, N+1);
        if(!file_date(FileName)) continue;
        Records = dataParse(1, Format, FileName); // read BBO data
        printf("\n%d rows read", Records);
        dataNew(2, Records, 2); // create T1 dataset
        for(Record = 0, Row = 0; Record < Records; Record++)
        {
            if(!strstr(Code, dataStr(1, Record, 1))) continue; // select only records
with correct symbol
            T1* t1 = dataStr(2, Row, 0); // store record in T1 format
            float Price = 0.01 * dataInt(1, Record, 3); // price in cents
            if(Price < 1000) continue; // no valid price
            string AskBid = dataStr(1, Record, 2);
            if(AskBid[0] == 'B') // negative price for Bid
                Price = -Price;
            t1->fVal = Price;
            t1->time = dataVar(1, Record, 0) + 1./24.; // add 1 hour Chicago-NY time
difference
            Row++;
        }
        printf(", %d stored", Row);
    }
}
```

```

        dataAppend(3, 2, 0, Row); // append dataset
        if(!wait(0)) return;
    }
    dataSave(3, OutName); // store complete dataset
}

```

We're converting here 10 days in October 2016 for our backtest. The script first parses the CSV file into an intermediary binary dataset, which is then converted to the T1 target format. Since the time stamps are in Chicago local time, we have to add one hour to convert them to NY time.

The company watching the New York Stock exchange delivers data in a highly compressed specific format named "NxCore Tape". We're using the Zorro NxCore plugin for converting this to another T1 list:

```

////////////////////////////////////
// Convert price history from Nanex .nx2 to .t1
////////////////////////////////////

#define STARTDAY 20161004
#define ENDDAY   20161014
#define BUFFER   10000

string InName = "History\\%8d.GS.nx2"; // name of a single day tape
string OutName = "History\\SPY_201610.t1";
string Code = "eSPY";

int Row, Rows;

typedef struct QUOTE {
    char    Name[24];
    var     Time, Price, Size;
} QUOTE;

int callback(QUOTE *Quote)
{
    if(!strstr(Quote->Name, Code)) return 1;
    T1* t1 = dataStr(1, Row, 0); // store record in T1 format
    t1->time = Quote->Time;
    t1->fVal = Quote->Price;
    Row++; Rows++;
    if(Row >= BUFFER) { // dataset full?
        Row = 0;
        dataAppend(2, 1); // append to dataset 2
    }
    return 1;
}

void main()
{
    dataNew(1, BUFFER, 2); // create a small dataset
    login(1);              // open the NxCore plugin

    int N;
    for(N = STARTDAY; N <= ENDDAY; N++) {
        string FileName = strf(InName, N);
        if(!file_date(FileName)) continue;
    }
}

```

```

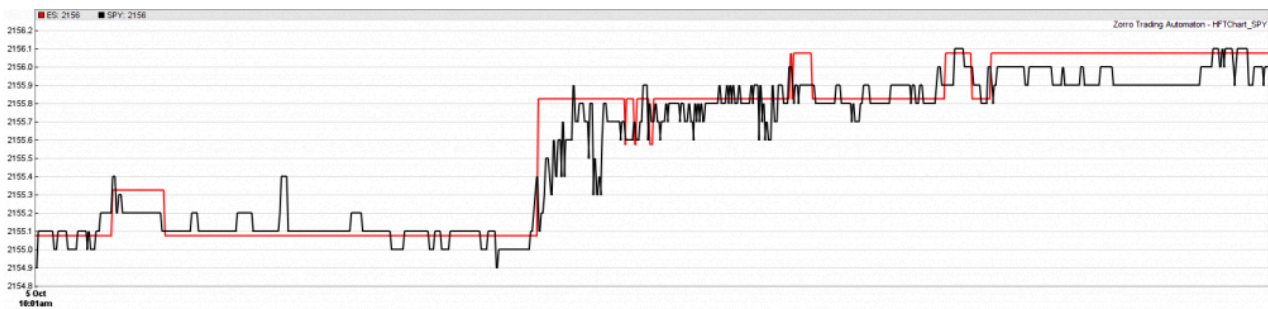
printf("\n%s..",FileName);
Row = Rows = 0; // initialize global variables
brokerCommand(SET_HISTORY,FileName); // parse the tape
dataAppend(2,1,0,Row); // append the rest to dataset 2
printf("\n%d rows stored",Rows);
if(!wait(0)) return; // abort when [Stop] was hit
}
dataSave(2,OutName); // store complete dataset
}

```

The **callback** function is called by any quote in the tape file. We don't need most of the data, so we filter out only the SPY quotes ("eSPY").

Verifying the inefficiency

With data from both sources, we can now compare the ES and SPY prices in high resolution. Here's a typical 10 seconds sample from the price curves:



SPY (black) vs. ES (red), October 5, 2017, 10:01:25 – 10:01:35

The resolution is 1 ms. ES is plotted in \$ units, SPY in 10 cents units, with a small offset so that the curves lie over each other. The prices shown are ask prices. We can see that ES moves in steps of 25 cents, SPY in steps of 1 cent. The prices are still well correlated even on a millisecond scale. ES appears to trail a tiny bit.

We can also see an arbitrage opportunity at the steep price step in the center at about 10:01:30. ES reacted a bit slower, but stronger on some event, probably a moderate price jump of one of the stocks of the S&P 500. This event also triggered a fast sequence of oscillating SPY quotes, most likely from other HFT systems, until the situation calmed down again a few 100 ms later (the scale is not linear since time periods with no quotes are skipped). For several milliseconds the ES-SPY difference exceeded the ask-bid spread of both assets (usually 25 cents for ES and 1.4 cents for SPY). We would here ideally sell ES and buy SPY immediately after the ES price step. So we have verified that the theoretized inefficiency does really exist, at least in this sample.

The script for displaying high resolution charts:

```
#define ES_HISTORY      "ES_201610.t1"
#define SPY_HISTORY     "SPY_201610.t1"
#define TIMEFORMAT     "%Y%m%d %H:%M:%S"
#define FACTOR          10
#define OFFSET          3.575

void main()
{
    var StartTime = wdatef(TIMEFORMAT, "20161005 10:01:25"),
        EndTime = wdatef(TIMEFORMAT, "20161005 10:01:35");
    MaxBars = 10000;
    BarPeriod = 0.001/60.; // 1 ms plot resolution
    Outlier = 1.002; // filter out 0.2% outliers

    assetList("HFT.csv");
    dataLoad(1, ES_HISTORY, 2);
    dataLoad(2, SPY_HISTORY, 2);
    int RowES=0, RowSPY=0;

    while(Bar < MaxBars)
    {
        var TimeES = dataVar(1, RowES, 0),
            PriceES = dataVar(1, RowES, 1),
            TimeSPY = dataVar(2, RowSPY, 0),
            PriceSPY = dataVar(2, RowSPY, 1);

        if(TimeES < TimeSPY) RowES++;
        else RowSPY++;

        if(min(TimeES, TimeSPY) < StartTime) continue;
        if(max(TimeES, TimeSPY) > EndTime) break;

        if(TimeES < TimeSPY) {
            asset("ES");
            priceQuote(TimeES, PriceES);
        } else {
            asset("SPY");
            priceQuote(TimeSPY, PriceSPY);
        }

        asset("ES");
        if(AssetBar > 0) plot("ES", AskPrice+OFFSET, LINE, RED);
        asset("SPY");
        if(AssetBar > 0) plot("SPY", FACTOR*AskPrice, LINE, BLACK);
    }
}
```

The script first reads the two historical data files that we've created above, and then parses them row by row. For keeping the ES and SPY quotes in sync, we always read the quote with the smaller timestamp from the datasets (the quotes are stored in ascending timestamp order). The **priceQuote** function checks the prices for outliers, stores the ask price in the **AskPrice** variable and the ask-bid difference in **Spread**, and increases the **Bar** count for plotting the price curves. A bar of the curve is equivalent to 1 ms. The **AssetBar** variable is the last bar with a

price quote of that asset, and is used here to prevent plotting before the first quote arrived.

Testing the system

For backtesting our HFT system, we only need to modify the script above a bit, and call the **tradeHFT** function in the loop:

```
#define LATENCY          4.0      // milliseconds

function main()
{
    var StartTime = wdatef(TIMEFORMAT, "20161005 09:30:00"),
        EndTime = wdatef(TIMEFORMAT, "20161005 15:30:00");
    MaxBars = 200000;
    BarPeriod = 0.1/60.;    // 100 ms bars
    Outlier = 1.002;

    assetList("HFT.csv");
    dataLoad(1, ES_HISTORY, 2);
    dataLoad(2, SPY_HISTORY, 2);
    int RowES=0, RowSPY=0;

    EntryDelay = LATENCY/1000.;
    Hedge = 2;
    Fill = 8; // HFT fill mode;
    Slippage = 0;
    Lots = 100;

    while (Bar < MaxBars)
    {
        var TimeES = dataVar(1, RowES, 0),
            PriceES = dataVar(1, RowES, 1),
            TimeSPY = dataVar(2, RowSPY, 0),
            PriceSPY = dataVar(2, RowSPY, 1);

        if (TimeES < TimeSPY) RowES++;
        else RowSPY++;

        if (min(TimeES, TimeSPY) < StartTime) continue;
        if (max(TimeES, TimeSPY) > EndTime) break;

        if (TimeES < TimeSPY) {
            asset("ES");
            priceQuote(TimeES, PriceES);
        } else {
            asset("SPY");
            priceQuote(TimeSPY, FACTOR*PriceSPY);
        }

        asset("ES");
        if (!AssetBar) continue;
        var AskES = AskPrice, BidES = AskPrice-Spread;
        asset("SPY");
        if (!AssetBar) continue;
        var AskSPY = AskPrice, BidSPY = AskPrice-Spread;

        int Order = tradeHFT(AskSPY, BidSPY, AskES, BidES);
    }
}
```

```

switch(Order) {
    case 1:
        asset("ES"); enterLong();
        asset("SPY"); enterShort();
        break;

    case 2:
        asset("ES"); enterShort();
        asset("SPY"); enterLong();
        break;

    case 0:
        asset("ES"); exitLong(); exitShort();
        asset("SPY"); exitLong(); exitShort();
        break;
}

printf("\nProfit %.2f at NY Time %s",
        Equity, strdate(TIMEFORMAT, dataVar(1, RowES, 0)));
}

```

This script runs a HFT backtest of one trading day, from 9:30 until 15:30 New York time. We can see that it just calls the HFT function with the ES and SPY prices, then executes the returned code in the switch-case statement. It opens 100 units of each asset (equivalent to 2 ES and 1000 SPY contracts). The round-trip latency is set up with the **EntryDelay** variable. In HFT mode (**Fill = 8**), a trade is filled at the most recent price quote after the given delay. This ensures a realistic latency simulation when price quotes are entered with their exchange time stamps.

Here's the HFT profit at the end of the day with different round-trip latency values:

| LATENCY | 0.5 ms | 4.0 ms | 6.0 ms | 10 ms |
|--------------|---------|---------|---------|--------|
| Profit / day | + \$793 | + \$273 | + \$205 | – \$15 |

The ES-SPY HFT arbitrage system makes about \$800 each day with an unrealistic small latency of 500 microseconds. Unfortunately, due to the 700 miles between the NYSE and the CME, you'd need a time machine for that (or some faster-than-light method of quantum teleportation). A HFT server in Warren, Ohio, at 4 ms latency would generate about \$300 per day. A server slightly off the direct NY-Chicago line can still grind out \$200 daily. But the system deteriorates quickly when located further away, as with a server in Nashville, Tennessee, at around 10 ms latency. This is a strong hint that some high-speed systems in the proximity of both exchanges are already busy with exploiting ES-SPY arbitrage.

Still, \$300 per day result in a decent \$75,000 annual income. However, what needed you to invest for that, aside from hardware and software? With SPY at \$250, the 100 units translate to $100 * \$2500 + 100 * 10 * \$250 =$ half a million dollars trade volume. So you would get only 15% annual return on your investment. But you can improve this by adding more pairs of NY ETFs and their equivalent CME futures to the arbitrage strategy. And you can improve it further when you find a broker or similar service that can receive your orders directly at the exchanges. Due to the ETF/future hedging the position is almost without risk. So you could probably negotiate a large leverage. And also a flat monthly fee, since broker commissions were not included in the test.

Conclusion

- When systems react fast enough, profits can be achieved with very primitive methods, such as arbitrage between highly correlated assets.
- Location has a large impact on HFT profits.
- ES-SPY arbitrage cannot be traded by everyone from everywhere. You're competing with people that are doing this already. Possibly in Warren, Ohio.

I've added the scripts and the asset list to the 2017 script archive in the "HFT" and "History" folders. Unfortunately I could not add the ES / SPY price history files, since I do not own the data. For reproducing the results, get BBO history from Nanex™ or Nanotick™ – their data can be read with the scripts above. You'll also need Zorro S version 1.60 or above, which supports HFT fill mode.

References

- (1) [When Correlations Break Down](#) (Jared Bernstein 2014)
- (2) [Financial Programming Greatly Accelerated](#) (Cousin/Weston, Automated Trader 42/2017)


Addendum. I have been asked how the profit would be affected when the server is located in New York, with 0.25 ms signal delay to the NYSE and 4.8 ms signal delay to the CME. For simulating this, modify the script:

```
var TimeES = dataVar(1, RowES, 0) + 4.8 / (1000 * 60 * 60 * 24),
    TimeSPY = dataVar(2, RowSPY, 0) + 0.25 / (1000 * 60 * 60 * 24),
...
switch(Order) {
  case 1:
    EntryDelay = 4.8 / 1000;
    asset("ES"); enterLong();
    EntryDelay = 0.25 / 1000;
    asset("SPY"); enterShort();
    break;

  case 2:
    EntryDelay = 4.8 / 1000;
    asset("ES"); enterShort();
    EntryDelay = 0.25 / 1000;
    asset("SPY"); enterLong();
    break;

  case 0:
    EntryDelay = 4.8 / 1000;
    asset("ES"); exitLong(); exitShort();
    EntryDelay = 0.25 / 1000;
    asset("SPY"); exitLong(); exitShort();
    break;
}
```

The first line simulates the price quote arrivals with 4.8 and 0.25 ms delay, the other lines establish the different order delays for SPY and ES. Alternatively, you could artificially delay the incoming SPY quotes by another 4.55 ms, so that their time stamps again match the ES quotes. In both cases the system returns about \$240 per day, almost as much as in Warren. However a similar system located in Aurora, close to Chicago (swap the delays in the script), would produce zero profit. The asymmetry is caused by the relatively long constant periods of ES, making the SPY latency more relevant for the money at the end of the day.

 jcl / July 8, 2017 / No Math, Programming, System Evaluation / Arbitrage, ES, ETF, HFT, Latency, NxCore, SPY

102 thoughts on “Hacking a HFT system”

**Alex**

July 10, 2017 at 01:05

Well done sir, great explanation

Pingback: [Quantocracy's Daily Wrap for 07/09/2017 | Quantocracy](#)

**Jochen Forer**

July 10, 2017 at 08:30

Very nice idea.

I wonder if it would be possible to use this kind of system to trade the arbitrage between different Bitcoin marketplaces and the connected other digital currencies. So far I know they bid a higher range of spreads between the different marketplaces and the different currencies in exchange to each other or in an normal Currency

**jcl**

July 10, 2017 at 08:55

Yes, and this could be much more promising since cryptocurrency trading is still at the pioneer stage and there are probably not many HFT systems trading bitcoins, if any at all. The problem is the infrastructure and reliability of bitcoin exchanges. You permanently hear that one or the other exchange is going down or does not pay out anymore.

**lorenzo**

July 11, 2017 at 14:37

U cant arbitrage bitcoin, there are too hight fee on exchange, look at blackbird, could do like 2% month, but now not even 0.5%

**Richard**

July 12, 2017 at 04:54

Hi,

Great article on HFT. I have one questions though. Could you elaborate on why you use SpreadSPY/SpreadES to decide the magnitude of the deviation in order to enter or close a position?

Thanks,

Rich

**jcl**

July 12, 2017 at 09:27

Because the price deviation must be greater than both ask-bid spreads for making profit. The ES spread is relatively constant, but SPY spread can be up to 6 cents, so it must be considered when opening a position. The spread is here split and checked on opening and on closing.

**Minsheng**

July 12, 2017 at 13:26

Nice work jcl. By the way, is the zorro script able to get price data from 2 brokers for the arbitrage?

**jcl**

July 12, 2017 at 14:17

Not this script, since it's for backtesting only. But generally, broker arbitrage should be no problem, I would use two parallel running Zorros for that. But I have heard that broker arbitrage rarely works.

**Jk**

July 12, 2017 at 15:53

This is a really nice article JCL. I enjoyed reading this immensely.

I have a lot of experience in these methods and your article took me back to the early days of this trade.

The main issue is the cost of the infrastructure to support this at a meaningful level. We used microwaves and lasers, which do not come cheap.

Sub 8ns can make a difference. While CME has equidistant cabling there are some other tricks you can employ to gain edge.

Great article.

**Jan**

July 12, 2017 at 20:29

Btw there is a book called Flash Boys written by Michael L. Lewis which describes the most common ways of HFT. There was also said that you can't just simply put your servers in the middle between two cities because the cable is not going straight – it goes zigzag through other cities so the best thing is to lay down your own cable.

**Tom**

August 2, 2017 at 10:40

Hi,
great tutorial there. Much appreciated 😊

How would I include the order fees for entering/exiting the position into the arbitrage calculations?

Lets say I have a takerFee of 0.025%,
can I just “increase” the spread like this for the system to work?
 $(askSPY + (askSPY * takerFee)) - (bidES - (bidES * takerFee))$



August 2, 2017 at 10:58

I would not add it to the spread. Either sum the fees up in a separate Fee variable, or subtract them directly from the predefined “Equity” variable that contains the accumulated profit from the trades.



August 12, 2017 at 06:53

you wouldn't use floats in HFT code . Too slow . Use two integers in a struct to represent a price , fixed precision . Additionally everyone is using microwave now for index arb between NY4 and Aurora.



August 12, 2017 at 07:44

HFT code today normally uses floats. Integer multiplication needs just about 1.4 clock cycles less than FPU multiplication, and since you have overhead for shifting etc, floats are often actually faster. Lite-C has a 22.10 fixed point variable type, but it gives no speed advantage on modern CPUs. You use integer only when you need it for special purposes, for instance realizing your system in hardware.

Pingback: [Hacking a HFT system – The Financial Hacker | Automated trading](https://financial-hacker.com/hacking-hft-systems/)



August 31, 2017 at 07:24

JCL – we do not use floating point of any type (doubles, floats) in HFT since it is very hard to compare. Fixed point is usually the choice.

Good piece. Some thoughts:

- (1) your SPY data was likely captured and timestamp'ed in NYC while the ES data was timestamp'ed in Aurora/Chicago.
- (2) most commercial firms that collect tick data for resale have really bad timestamps, which drift even milliseconds intraday.
- (3) Most companies (eg Reuters) capture all data in just one location (eg NYC) even if the exchange is in Chicago – and they use the cheapest network – that would be very favorable in a backtesting
- (4) ES futures have a discount in relation to spot SPY depending on the amount of days until expiration that you are not taking in account;
- (5) You are crossing the market and paying the spread every time you trade – you are not accounting for it; Every time you sell a stock or future you have to use the BID/lower price and vice versa.
- (5.a) You might mitigate the spread-crossing by using a market making simulator that will take in account market volume and queue position but then you will have to code also a more flexible position management. In this case the trading logic will be more complex too.
- (6) you are not accounting for fees and commissions.
- (7) VHDL for FPGA is still software and most RTL development is done in Verilog in my experience, not VHDL.
- (8) FPGAs nowadays have a roundtrip of 200 nanos, tick-to-trade, including trading logic.
- (9) you are not checking if the book spread is wide – this happens frequently and throws off any EMA.



August 31, 2017 at 11:41

FP compare problems should play no role since you normally compare floats on above or below, not on equality. I can imagine the use of fixed point in some special cases, such as counting discrete price steps, or when your system runs in a FPGA.

Timestamps from data collecting firms are not used in HFT testing. You're using exchange timestamps. Expiration discount is not very relevant when your trades last milliseconds. Market spreads however are relevant, and are of course accounted for as you see in the scripts. If I would disregard market spreads, the script would make ten thousands of dollars every day, not a few hundred. Fees are not included as already mentioned. They normally depend on your infrastructure and not on the HFT algorithm. VHDL and Verilog are both languages for defining hardware components. We use VHDL, but that's a matter of personal preference.



September 10, 2017 at 05:54

Jcl,

Thanks for all the info, it is very educational!

I have just one question I can't help but wonder about:

As someone who can charge EUR170 for changing just one line of code sometimes, writing this extensive public blog must be worth a fortune in your time. Are you paid to do it? (Since you mentioned that you'd be ready to make youtube lectures for pay, I assume you're paid for the blog). And more importantly, what is the agenda of the entity who pays you? What do they gain from it? ...unless it's a secret of course.



September 10, 2017 at 07:23

Just as most bloggers, I'm doing this in my free time, so there's no sinister entity that pays me. And although we theoretically indeed must charge EUR 170 for changing a single line of code, in reality this doesn't happen. In such a case we advise the client how to change that line by himself.



Trader123

September 10, 2017 at 17:10

Oh I see, but in that case you might want to consider making a youtube video blog after all. The video medium is arguably more interactive and engaging. It would greatly enhance your altruistic effort of communicating your experience and ideas to the general public. I know I would watch your videos for sure.



Kagaratsch

September 11, 2017 at 05:34

Hi JCL,

Thanks for all your hard work!

I am currently reading your book, halfway through and feel like I'm learning a lot. However, there is one statement at the beginning (or at the "begin" as you like to say :)) that got me confused: You said there that "higher leverage does not increase risk, and in fact the higher the leverage the further we are from a margin call".

I just cant wrap my head around that statement. I always thought that leverage multiplies (amplifies) the volatility of an asset in relation to our margin. Let's say a hypothetical un-leveraged price movement in some position of ours that goes against us claims 10% of our margin – then we'd only be wiped out by 10 consecutive movements against us (our stop loss would have time to react).

However, leveraging e.g. 10x times would in the same situation make our whole margin disappear within that one price movement instead of ten. Which means risk is indeed amplified and we are much more exposed to the margin call danger from simple random fluctuations. Am I getting something wrong?

September 11, 2017 at 08:20

I probably should have said: Higher leverage does not increase risk **with the same trade volume**.

If you trade a volume X , you get a margin call when your loss exceeds your capital minus allocated margin, which is $X/\text{Leverage}$. You see that the higher the leverage, the bigger is the term $(\text{Capital} - X/\text{Leverage})$ and thus the bigger is the maximum loss that your account can survive.

Higher leverage does in fact increase the risk when it tempts you to trade a higher volume.



Kagaratsch

September 11, 2017 at 13:41

Ohha! So, we are actually always buying the same amount of asset, except when we leverage we have to commit less margin to back the transaction. Which in turn means we have less margin at risk! That make sense, thank you for the explanation.



Kagaratsch

September 14, 2017 at 02:15

So I finished reading the “Black Book of Financial Hacking” and went over to study the Zorro manual – and what do I see? The book and the manual have 90% overlap!! And the manual actually goes far more in depth. Consensus for the reader: Don’t waste your money on the book, just read the manual.

JCL, as the author of the book and the Zorro manual would you agree with this observation?

Considering that the Zorro project is sponsored you already must have earned good money from working on the manual. And so, would you testify to be practicing a shameless cash grab by reselling a shortened version of the manual as your book? xD hahahaha

**Min**

September 14, 2017 at 07:25

I bought the book too. I think the book is very useful in explaining the fundamental concepts more than manual. I did not regret buying the book. I would still recommend it to beginners.

**jcl**

September 14, 2017 at 11:53

Just to clarify, even though I wrote both the book and the manual, they are completely different. One is an introduction in algo trading and the other is a software manual that assumes that you know algo trading already. The only overlap is the programming course, but for obvious reasons a software manual can not provide as many details and examples as a book. The course in the manual has 7 pages and the course in the book maybe 100 pages.

**Kagaratsch**

September 14, 2017 at 13:53

I dunno, from what I've seen even the motivating conversations between Alice and Bob for each exercise are exactly the same... But maybe that is just my impression. ^^

**AndrewAMD**

September 14, 2017 at 18:03

jcl, off-topic, but I see that you will be presenting at Quant Expo. I will be unable to attend. Will you be recording the presentation by any chance?

**jcl**

September 15, 2017 at 09:46

I believe it will be recorded by the organisator, but I don't know where and when the video will be published.

**Jon**

November 21, 2017 at 00:30

Slightly off-topic but on the idea of HFT. Is it possible to arbitrage spread from a retail account?

Place both Buy and sell orders to ignore price movements and just focus on difference in spread?

**jcl**

November 21, 2017 at 09:10

Sure, this is called statistical arbitrage. You need 2 assets that are highly correlated, but a bit less correlated than ES and SPY.

**BeginnerTrader101**

January 1, 2018 at 20:20

Continuing on last comment...

How is statistical Arb possible for a retail investor (it is so precisely time bound). Manually it looks impossible to me – <http://www.nasdaq.com/article/dont-be-fooled-by-the-fancy-name-statistical-arbitrage-is-a-simple-way-to-profit-cm254669>

Quite interested to know your views. And thanks for such a great blog !!

**jcl**

January 3, 2018 at 16:27

Our HFT clients were not really retail traders. But stat arb is easily possible, you just normally use other pairs where the deviations are long-termed and precise timing is less relevant. Millisecond trades and colocating servers close to the exchange is indeed a bit difficult for private traders, although I had heard of some who allegedly did that.

 **Alex**

January 22, 2018 at 17:13

Sorry, is the 2017 script archive corrupt? Thx

 **jcl** 

January 22, 2018 at 17:33

No, what's the problem? I've just tested it and can download and open it.

 **mitch tullman**

January 22, 2018 at 18:38

Sir,
Do you remember how many trades you did during your test in 2016?

 **jcl** 

January 23, 2018 at 10:22

No, but it was not many, maybe 2 or 3 per day.

Pingback: [zulutrade discount code for renewal – zulutradepro](#)

Pingback: [Hacking a HFT system | Samuelssons Rapport](#)

**Jeferson A Silva**

May 22, 2018 at 21:14

With these parameters is it possible to create an HFT theft?

I confess it is my biggest dream, to have software of this greatness, but I do not have the knowledge to create it.

**jcl**

May 23, 2018 at 10:05

The method above is legal trading, no theft. It would be theft when you used such a system to send fake orders for manipulating the order book. This would be, of course, illegal.

**Alex**

May 24, 2018 at 00:26

> the closer to the main network hub, the dearer the space.

No matter how close you are to the main network hub in exchange data center – the length of the cable from you to trading engine will be the same – that's the law imposed to exchange by securities comission

And by the way, hft are not using fore-running

**jcl**

June 3, 2018 at 16:32

Yes, I learned that all US exchanges have meanwhile equidistant cabling. That was different in the past.

**Daniel**

October 11, 2018 at 00:27

ES is \$12.50 a tick.



Peter Williams

October 12, 2018 at 11:59

Interesting.

My Black Book is currently in transit and looking forward to reading the information – despite some negative comments above.

Is it possible to do something similar with FOREX?

Maybe just gaining a little of the edge the 'big boys' have could possibly enable an excellent opportunity to get on a 'movement' just ahead of the conventional broker with MT4 platform.



jcl

October 12, 2018 at 12:09

Broker arbitrage is theoretically possible with Forex, but difficult because Forex brokers are doing all they can for preventing it.



Rajeev

October 12, 2018 at 19:01

I am reading your book currently and do not understand your statement “higher leverage does not increase risk, and in fact the higher the leverage the further we are from a margin call”. How can having a higher leverage further the margin call? Lets take a simple example, say I am buying one share of stock XYZ at \$100. I can do it two different ways,

Case 1) I borrow \$50 from the broker so my equity is \$50 and the leverage ration will be 2:1

or

Case 2) I borrow \$80 so my equity is only \$20 and the leverage ration will be 5:1

If the price of the stock falls from \$100 to \$80 then I will get a margin call in case (2) but not in case (1) so I don't understand your statement stating higher leverage ratio which is case (2) makes margin call further. Am I missing something?

 **jcl** 

October 13, 2018 at 08:58

Possibly. When you have \$100 on your account, borrow \$80, and pay \$20, then you have \$80 cash remaining – at least with the math I learned in school. A \$50 drop will blow your account in case 1, but not in case 2.

 **Rajeev**

October 13, 2018 at 14:52

Maybe I was not clear, in case (1) I start with \$50 and borrow another \$50 to buy a share of XYZ. In case (2) I start with \$20 and borrow another \$80. Now when the price of the stock falls from \$100 to \$80 then I will get a margin call in case(2) but not in case (1). Do you agree?

 **jcl** 

October 13, 2018 at 15:53

Certainly. Having more money on an account is an even more brilliant method for avoiding margin calls, than higher leverage.

 **Rajeev**

October 14, 2018 at 18:33

Since you agree with my comment above doesn't it imply from that example that lower leverage is better than higher leverage in avoiding the margin calls?

**jcl**

October 15, 2018 at 09:18

Hmm. If I had known that leverage is so difficult, I had probably given some more examples in my book.

Last try. Contemplate this. You have two accounts, and put on both exactly the same capital, and buy exactly the same number of the same stocks. Only the account leverage is different. The stock tanks. Which account will get the margin call first, with the low or with the high leverage?

It might help when you get a pen and paper, enter some numbers for the capital and leverages, and do the math.

**Rajeev**

October 15, 2018 at 15:19

I think the reason for this confusion is because I assumed you were referring to leverage used in an account but I think you are referring to the maximum leverage allowed in an account. If we assume you meant maximum leverage allowed in an account, then yes the account which allows higher leverage is further from a margin call than the one with lower leverage for the same trade volume. Agree?

**jcl**

October 15, 2018 at 15:40

Yes.

**Lucas**

October 18, 2018 at 02:28

Amazing article JCL! Super useful. Thank you!

I am interested in what your take is on implementing a HFT system in a market that isn't very liquid (as is the case in my country). The forex spot and futures markets are the most active with anywhere between 1 or 2 seconds, to minutes between transactions. What considerations would you make to better adapt the system to this market? Is there a strategy in particular that is better suited for this case?



October 18, 2018 at 08:50

Long term systems can work well in an illiquid market, but short term and HFT can be a bit difficult when quotes are rare and you cannot really predict at which price your order will be filled.



October 19, 2018 at 14:19

Any suggestions for building models for an illiquid market? Or any resources where I could learn this. I'm really set on building the best system I can, but I have a lot to learn. Any help or advice you could provide is greatly appreciated.

Also, is BBO data really necessary or is historical data of every transaction okay? I figured BBO data would be better since the spreads are an important factor in an illiquid market.



October 22, 2018 at 15:47

Yes, BBO is better. The less liquid the market, the further trading results will deviate from a theoretical ideal and the more important is a precise backtest that includes spread and volume.



January 18, 2019 at 10:58

This post is fabulous and eye opening.

I read from the comments that it will be tough to arbitrage Forex between brokers.

How about arbitrage indices between brokers for retail traders? Will this be easier?



Dan

February 27, 2019 at 18:34

Jcl I have a really dumb question if hft systems require loads of data to operate and since you can't have them all at the exchange but you also can't separate them for the sake of latency how does the hft infrastructure work I'm positive i just have a terrible grasp of this



jcl

February 28, 2019 at 10:39

Live HFT systems normally require no historical data because they use no technical indicators. But if they did, it's no problem to store it. Having loads of data on the machine won't make it slower.



Dan

March 1, 2019 at 09:27

Thanks jcl for the response as you can see my lack of knowledge on this topic can you please explain to me layman's terms how the setup of a live hft system works and possibly point me to sources for more clarity



jcl

March 1, 2019 at 09:41

At which exchange or which colocation service do you want to run your HFT system?

 **dan**

March 1, 2019 at 10:02

Let's say nyse

 **Dan**

March 1, 2019 at 10:05

So my current line of thinking is that you have petabytes of data you develop a profitable model using that data then you put that algorithm in that server place the server at exchange and it trades that algorithm without interacting with the petabytes of data you used

 **jcl** 

March 1, 2019 at 10:28

I don't know if you really need petabytes of data. Hopefully terabytes are sufficient. But yes, that's basically the procedure. For the setup, ask the NYSE data center.

 **Dan**

March 1, 2019 at 11:15

Thx, and I mention petabytes because of big hft firms

 **Eric Jiang**

April 5, 2019 at 10:18

I learn a lot from this article. thanks.

**Dave McCrory**

June 15, 2019 at 06:12

Great article, explained quite a bit and increased my understanding. I was wondering when you show the values of .5ms = \$793 and 4ms = \$273, etc. What is the entire spread? meaning 1ms, 2ms, 3ms... I'm curious of what the drop-off looks like from ms to ms.

**jcl**

June 15, 2019 at 12:29

I don't have the exact data anymore, but the profit dropped sharply above 0.5 ms and then stayed in the \$200-\$300 area over a relatively long latency range.

**Dave McCrory**

June 15, 2019 at 22:53

Valuable insight nonetheless. Do you consider this typical behavior for arbitrage or is every situation 100% unique or are there maybe categories of behavior depending on what is being arbitrated. If there are categories, what would they be? I'm asking all these questions because I'm doing research (for myself).

Thanks again!

**jcl**

June 17, 2019 at 08:43

From my experience, every strategy is unique, just as in non-HFT systems. So the results cannot be generalized. You must really test your strategy with different simulated latencies.

**Montana**

July 13, 2019 at 06:12

Is this actually still doable? Because as another user said above “the length of the cable from you to trading engine will be the same” so you can’t bank on the shorter latency am I right? And let’s just say I wanted to do this as a retail trader, would I do it through my broker’s API or do I have to set up my own “firm” with the SEC to execute my own orders?

**jcl**

July 13, 2019 at 16:49

The method described here uses no colocation at the exchange. But if you want to put your PC on a rack in the NYSE cellar, you’ll indeed get the same cable distance to the router as all the other PCs there. Or so they claim.

**Chester Doraemon**

April 15, 2020 at 14:31

I don’t think you’ve ever done the trade you talk about. It’s a lot more complicated than what you mentioned. There is a basis amount and you also don’t check that the liquidity is even there, that you need 5 SPY’s for 1 ES. Furthermore, ES is over 100k a contract so basis risk matters as does accurately managing funding rates and the SPY dividend.

Lastly, there is no point putting the server in Warren because the trade signal still needs to be sent out to the match engine. So you still have to cross the same distance from tick 2 trade.

from a former index trader

**jcl**

April 15, 2020 at 15:30

Thank you for sharing your opinion, and I'm impressed that you know the contract sizes of ES and SPY. But even when not understanding the system details, maybe you can imagine that an arbitrage test would not produce any result when the "basis amount" were wrong. Not even on a server in Warren.



Filimon George

May 28, 2020 at 14:49

Hello!

How can I implement that in an indicator?



jcl

May 28, 2020 at 16:04

You would normally encapsulate the algorithm in a function. But it is no indicator.



Cepturion

July 28, 2020 at 10:03

I could get a Server in LD4 and NY4 (LMAX UK connection to LD4 would be 0.4ms – LMAX US connection to NY4 about 0.45ms)

Is there a way to profit from HFT?

Maybe something like arbitrage?

Would be very interesting to work together maybe



jcl

July 28, 2020 at 16:25

I'm just posting the answer from Zorro support:

Arbitrage between London and New York is theoretically possible, for instance with US and EUR ETFs of the same index. But you would need a fast enough connection between LMAX UK and LMAX US, or from your server to both.



mitch tullman

August 23, 2020 at 17:22

I recently hired a coder to try to replicate your results. We got to the point where we tested using my IB data feed (which proved to be way too slow). To explore different feeds I would have to pay for coding to different API's (coder already had API to IB coded).....My question to you iswhat data feed could a regular, independent trader like myself have access to that would be fast enough??



jcl

August 23, 2020 at 17:40

IB is indeed too slow for HFT arbitrage. But there are services that colocate close to the CME or the NYSE, and offer transit times less than one microsecond. I can't advertise a particular service here, but you'll find them on the Internet.



Mitch

August 23, 2020 at 20:18

Thank you, but aren't they prohibitively expensive given our modest P&L assumptions here ?



jcl

August 24, 2020 at 10:59

Depends on your trade volume, but most are in reach for private traders.



luca

September 9, 2020 at 20:47

Hi jcl,

the prospect of arbitrage on SPY and ES seems interesting but verifying the costs of accessing the realtime data of the CBOE seem prohibitive for a frequency of a few milliseconds. One minute is equal to 60,000 milliseconds, so to work with the proposed strategy I would have to make over 10,000 calls, exceeding the rate limit per minute of 7500 among other things. There is a more efficient way to retrieve data in real time from the CBOE ? thank you

 jcl

September 10, 2020 at 09:39

I don't know the details of the CBOE connection service, but for HFT arbitrage you need direct access to the market without a rate limit.

 luca

September 21, 2020 at 21:21

would you have a suggestion from which service to retrieve the HFT data for an algorithm backtesting? thank you

 jcl

September 22, 2020 at 11:09

Often used are Nanotick and Nanex.

 luca zinato

September 23, 2020 at 07:21

based on your experience which broker i could try to use with nanex + zorro that has good latency? thank you so muc



September 23, 2020 at 07:37

It depends on what latency you need and what you want to pay for the service. We can't recommend a particular one. Check their advertised latencies and prices, and compare.



November 11, 2020 at 07:41

Hi JCL,

thanks for the great content. I have tried to implement above's approach as well as the Simple Broker Arbitrage example from the manual in the Zorro platform using two MT4 Terminals with Zorro plugin.

I was wondering why – when trying to print the ticks received from the 'function tick()' – Zorro tends to print multiple lines for the same timestamp, sometimes with differing prices sometimes with the same prices. I am receiving the prices as in the Simple Broker Arbitrage example via 'asset("...")'.

```
06:31:36.662 13164.5 13170.7 13164.6 13170.6
06:31:36.662 13164.5 13170.7 13164.6 13170.6
06:31:44.784 13166 13172.2 13166.1 13172.1
06:31:44.784 13166 13172.2 13166.1 13172.1
06:31:44.784 13166 13172.2 13166.1 13172.1
```

I hope my question is not too unrelated to this article. I have spent quite some time trying to figure out what I am doing wrong and would be very happy if you could shed some light.

Thanks a lot

Tom



November 11, 2020 at 14:54

MT4 terminals are not suited for HFT arbitrage, but I think you know that. The reason of the multiple ticks in a backtest is likely that these ticks are also repeated in the historical data – you can check the data with the History script. If there are multiple ticks, there's a dataparse flag that can be used to filter them out.



November 15, 2020 at 16:47

I was referring to live data not historical data. Zorro prints multiple/duplicate ticks with the same timestamp when using below's code to receive tickdata from two MT4 terminals (this is basically the code from your Simple Broker Arb example).

```
function tick()
{
asset(ASSET_A);
var AskA = priceClose(), BidA = priceClose() - Spread;
asset(ASSET_B);
var AskB = priceClose(), BidB = priceClose() - Spread;

printf("\n[%s.%0f],%.2f,%.2f,%.2f,%.2f,%.i",strdate(HMS,0),1000.*modf(second(),0),BidA,AskA,BidB,AskB);
}
```

I know MT4 is too slow for actual HFT trading, but still it would be good to know that tickdata received from MT4 via Zorro is reliable and that there is not something wrong. Though, most likely I am the one doing something wrong here ;).

I hope this is not leading too off topic. I would very much appreciate hearing back from you!

Thanks

Tom



jcl

November 16, 2020 at 15:40

MT4 ticks have no exchange timestamps. That means the time stamps that you see are the time when Zorro received them. Because the MT4 plugin is not streaming but polled, multiple ticks can come in at the same time. This happens when either Zorro's polling cycle or the MT4 polling cycle took longer than the time between the ticks. For Zorro you can set up the cycle with the TickTime variable, for MT4 there's a setting in the EA code. I believe it's 20 ms by default. It cannot be set much smaller because MT4 will then start choking.



Tom

November 17, 2020 at 12:15

You are mentioning the Zorro polling cycle as well as the MT4 polling cycle. Does this mean that the MT4 client actually polls the data from the broker's server or is it being pushed by the server?

Do you happen to know how the MT5 terminal handles this topic? Is the tickdata being pushed to the MT5 client and subsequently to the Zorro client or is there polling happening as well?

Thanks a lot for the sharing such informative details!



jcl

November 17, 2020 at 15:20

I don't know if MT4/5 poll data from their servers, but they use polling to pass the data to their EAs or to other platforms.

**William**

February 13, 2021 at 20:51

Hi jcl,

So bad that I have found your article this late. As long as I was looking for more answers, I googled “FPGA metatrader4”.

Anyways, this article/blog/study is very interesting.

“Broker arbitrage is theoretically possible with Forex, but difficult because Forex brokers are doing all they can for preventing it.”. What do you mean by that? How would they avoid to follow real price action?

So actually, I am interested in this arbitrage thing and think that the only real way to address arbitrage is FPGAs and SoCs and obviously coupled to a very low latency. I have started a little work in a first place to define whether a strategy can be applied and which should be the brokers to choose to apply it. I have published it on this forum (<https://www.forexfactory.com/thread/post/13314153>) but I am facing the nasty obstacle of the 15ms resolution, I will probably work out later.

JCL, do you have any idea whether an FPGA/SoC can directly connect to a MetaTrader broker server without using MT4 or MT5?

Thank you for work anyway.

**jcl**

February 14, 2021 at 13:06

Thank you. I think forex brokers prevent broker arbitrage mainly by analyzing suspicious trading behavior, and by their business terms. They might kick you out when they suspect you of broker arbitrage. Usually they notice it after a couple days and the arbitrage opportunity will then disappear.

For connecting a FPGA based system to a MT4 or MT5 server you need to analyze their traffic with the terminal. I don't know if they have a documented API for connecting to their server, but I doubt it.

 **William**

February 18, 2021 at 03:12

Hi thank you for your detailed answer. Well, regarding broker arbitrage policy, I already heard stuff like that but didn't really give a glance at the rules. I hope they are not dishonest enough to kick you out with neither notice, nor mention in the agreement...

As far as I am concerned, there may be 2 main arbitrage techniques both require a very low latency and a very fast system (that is the where the FPGA could intervene even if there is more to take with a latency than a FPGA over a PC system) :

Latency arbitrage which makes you take opportunities on a slow broker with signals coming from a faster broker. The ideal condition would be that all the brokers' prices were exactly the same but with a little time discrepancy between them. As you will have seen in my pictures on the link, this is not that simple... Sometimes a broker shows a price that evolves in a clear direction and the other one doesn't follow and sometimes they have the same behaviour. So it becomes complicated to infer from this an algorithm that would offer you a 99% winrate. In this case, maybe the AI may give a good help.

The other form of arbitrage that could be used on retail trading systems is triangular arbitrage for which for a very short period of time you will have two of three correlated currency pairs (let's say EURUSD/GBPUSD/EURGBP) that offers an opportunity .

But when trade on PC, with MT4 , you can't take arbitrage opportunities fast enough, whereas they say MT5 is suited for HFT and arbitrage techniques. The FPGA (SoC) would be a very good assistant but as you said, we would need the API.

Best regards

This site uses Akismet to reduce spam. [Learn how your comment data is processed.](#)

The Financial Hacker / Powered by WordPress / (c) by Johann Christian Lotter

