股票数据格式文档

更新日期：	2011年04月09日
网站：	http://alantop.5166.info
作者：	alantop
版本号：	2011.04.09

本文档用于揭示现存主流股票分析软件的数据存盘格式。

股票数据的应用如下:

1. 用于自己写选股器，完全按照自己的选股策略，完全摆脱原有股票软件的限制。
2. 用于研究股票数据格式，通过对比你可以看出来，那种数据格式效率更高。
3. 用于股票软件开发。

4.

大智慧数据格式

大智慧新一代目录结构

主目录 下的文件 :
  SYSN.DTA 存放系统提供的公式
  *.xml 动态显示牌不同页面布局文件
  *.htk 快捷键文件
USERDATA 下的文件:
  AUTOBLK.CFG: 自动板块设定
  SELF.DTA 存放用户自编的公式
BLOCK 文件夹下的文件:
  *.IBK 板块指数定义
  *.BLK 板块定义
  *.EBK 条件选股结果
SELF 文件夹下的文件:
  *.WSV 保存页面文件
  ALERT.DAT 历史预警纪录
  EXTDATA.INF 扩展数据定义
  *.CEP 保存组合条件选股条件
  TEMPCMPD.CEP 测试附加条件
  *.INV 用户个人投资纪录
  *.TPT 保存指标模板
  SELF 年月日 .DTA 每日自动公式备份文件

TEST 文件夹下的文件:
  *.TST 存放系统测试结果
  *.OPT 存放参数优化的结果
PARAM 参数指引文件夹

*.PRM 存放参数指引的结果
TABLE 文件夹下的文件：
*.ESS 数据表文件
*.ESD 数据表文件（带数据保存）
SelfData 文件夹下的文件：
*.str 字符串数据

*.oth 与股票无关序列值数据
Pattern 文件夹下的文件
*.PIN 模式匹配设计
*.PWT 模式匹配方法
S potAna 文件夹下的文件 ：
*.SPT 定位分析结果
Relate 文件夹下的文件 ：
*.RTL 相关分析结果
Posible 文件夹下的文件 ：
*.PSB 预测分布设计
DATA 件夹下的文件：
DAY.DAT 日线数据
EXTDAY.DAT 扩展数据
MIN.DAT 5 分钟线数据
REPORT.DAT 当天的分笔成交数据
STKINFO.DAT 代码表 / 即时行情数据 / 财务数据 / 除权数据
*.PRP 历史回忆数据，一天一个文件
abh.txt A 股、 B 股及 H 股相关信息文件
NEWS 文件夹下的文件：
*.TXT 财经报道、上交所公告、深交所公告

大智慧新一代 2.08.07.0907日线数据格式

大智慧二和大智慧三的日线数据格式没有变化。这个文档适用于c++开发人员。
以深圳为例 数据放在 C:\dzh2\data\sz\day.dat

数据结构共分三个结构体：

1．开始部分，存放一些全局数据的结构体（共24个字节）
由0x00 - 0x17开始

起止地址 数据内容 数据含义 数据类型
00 - 03 F4 9B 13 FC 日线文件标志 Integer
04 - 07 10 02 00 00 保留 Integer
08 - 0B 00 00 00 00 保留 Integer
0C - 0F D1 04 00 00 证券总数 Integer
10 - 13 81 0C 00 00 需添加之起始块号 Integer 就是文件的最后，计算方法是 0x41000 + 这个数字 * 8192
14 - 17 48 0C 00 00 当前最后空块号 Integer 最后一个空块 方法同上，就是说写数据，就在这个地方写，写完就在上面地方新增加新的块

```
struct GLOBAL_DATA
{
    int dayflag;
    int reserve1;
```

```
    int reserve2;
    int stocksum;
    int startblock;
    int lastblock;
};
```

2．存放后面具体数据的索引 (64个字节)
由0x18 - 0x58开始，总共有多少由0x0c提供.

从18h开始至40017h每64byte为一条股票数据分配记录，含义下表18h - 57h所示

起止地址 数据内容 数据含义 数据类型
18 - 21 31 41 30 30 30...FF 证券代码 Char[10]
22 - 25 B0 09 00 00 日线记录数 Integer
26 - 57 16 00 17 00...FF FF 记录块号 Word[25]

```
struct INDEX_DATA
{
        char code[10];
        int dayrecordnum;
        unsigned short int reocrd[25];
};
```

3．具体数据的内容(32个字节)
0x41000开始

8192 = 256 * 32

从41000h开始每8KB为一股票数据存储块，每个股票数据存储块共存储256条日线记录，每一条记录的长度为32 byte（含义如上表；上涨家数及下跌家数只对指数有效）；
从41000h开始的8KB为第0号数据存储块，以后类推；
系统对每个股票日线数据存储以存储块为单位进行分配。
日期字段的意义为：实际日期 = StrToDate(^1970-01-01^)+(日期字段 div 86400)；即"日期字段"除以86400所得数为实际日期距1970年01月01日的天数。
对于c++来说直接用__time32_t类型直接出来就是日期(vs2005中，这个代表32位时间类型值，在vc6中直接用time_t即可【vc2005中time_t代表64位时间值】)

起止地址 数据内容 数据含义 数据类型
41000 - 41003 80 47 B2 2B 日期 Integer
41004 - 41007 B9 1E 25 41 开盘价 Single
41008 - 4100B CD CC 4C 41 最高价 Single
4100C - 4100F EC 51 18 41 最低价 Single
41010 - 41013 9A 99 41 41 收盘价 Single
41014 - 41017 80 06 B2 47 成交量 Single
41018 - 4101B 40 1C BC 4C 成交金额 Single
4101C - 4101D 00 00 上涨家数 Word
4101E - 4101F 00 00 下跌家数 Word

struct DAY_DATA

```
{
__time32_t date;
float open;
float high;
float low;
float close;
float amount;
float money;
unsigned short int rise;
unsigned short int fall;
};
```

大智慧版块数据格式

关键的数据文件plank.ctf,plank.cxt。
其中ctf文件存放的是一个目录，是平行结构的。
而cxt文件则是存放具体板块下的股票代码数据。

格式分别为：
ctf格式：
该文件最前面的124个byte数据最好是舍弃不要。因为首先是基本上没有数据，其次，格式似乎跟后面的有些出入。从125个byte开始，每18个byte为一个板块的数据。
0~1 为某板块的股票个数
2~9 为该板块的名称（中文）
10~13 为该板块的股票的起始位置，该位置与cxt中对应。后面会详细介绍。
14~17 为该板块的股票的截止位置。同上。
该文件中间部分存在着很多无法正确获取名称的板块，但格式与这里是完全一致的。

cxt格式：
该文件，每12个byte为一个股票的数据。
0~3 为该股票的位置。此位置与ctf中的位置完全对应。
4~9 为该股票的代码
10~11 。

读出来后，ctf举例读出
板块名称 股票数 起始位置 截止位置
准权证 3 6545 6547
cxt中可以对应的找到
股票代码 位置
000752 6544
600873 -1(即那堆FFFF)
600688 6546
600795 6547
600028 -1
此时，从600688开始一直到600028就都是准权证板块的了。即起始位置所标注的下一个股票，截止位置所标注的下一个股票。就好了。这样通过这两个表的相互配合，就可以完成所有板块以及相关股票的对应关系了。

大智慧日K线的数据结构

一、数据文件和数据结构：
大智慧数据文件和数据结构：（假设大智慧股票行情软件安装在D:dzh目录下）
上海日线存储路径为:D:dzhDATASHaseDay,文件扩展名为:.day
上海周线存储路径为:D:dzhDATASHaseweek,文件扩展名为: .wek
上海月线存储路径为:D:dzhDATASHasemonth,文件扩展名为: .mnt
深圳日线存储路径为:D:dzhDATASZnseDay
深圳周线存储路径为:D:dzhDATASZnseweek
深圳月线存储路径为:D:dzhDATASZnsemonth
周线,月线格式与日线格式一致.
以深发展日线为例：

```
1A76:0100 D6 CD 2F 01 52 07 01 00-52 07 01 00 52 07 01 00
1A76:0110 52 07 01 00 86 0F 00 00-4D 02 00 00 00 00 00 00
1A76:0120 00 00 00 00 00 00 00 00-D7 CD 2F 01 60 03 01 00
1A76:0130 60 03 01 00 60 03 01 00-60 03 01 00 82 05 00 00
1A76:0140 D4 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
```

每一条记录的长度为40字节：
1-4字节为日期,D6 CD 2F 01转换为十进制是:19910102
5-8字节=开盘价(元)*1000
9-12字节=最高价(元)*1000
13-16字节=最低价(元)*1000
17-20字节=收盘价(元)*1000
21-24字节=成交金额(元)/1000
25-28字节=成交量(手)
其余12字节未使用
实现步骤：
1、先定义日线数据结构数组
2、再以实际记录数分配动态数组空间
3、然后把数据读入相应数组中

日线数据放在：%app_dir%\DATA\SHase\Day（上证A股） 以及 %app_dir%\DATA\SZnse\Day（深圳A股）

```
#pragma once

struct DZH5Day
{
unsigned long date;//date的格式：20070423
unsigned long open;//开盘价
unsigned long high;//最高价
unsigned long low;//最低价
unsigned long close;//收盘价
unsigned long moneysum;//成交金额
unsigned long turnover;//成交数量
char unused[12];//保留
};
```

整个结构共40个字节，读者可以查看所有的日线文件，大小肯定是40的倍数

这样很容易读出相应的数据：
如：

```
FILE* pFile = fopen("600001.day", "rb");
if (NULL != pFile)
{
while(!feof(pFile))
{
DZH5Day dayK;
fread(&dayK, sizeof(DZH5Day), 1, pFile);
......
}
}
```

大智慧Internet版 5.58.2760 的数据格式

上海日线数据目录： C:\dzh\DATA\SHase\Day
深圳日线数据目录： C:\dzh\DATA\SZnse\Day

每个数据块40个字节。

每个文件一开头就是日数据，不像有些股软数据开始有一些格式。
每日数据一共为40个字节。
第一个四字节：日期，转换为十进制即可。
第二个四字节：开盘，除以1000
第三个四字节：最高，除以1000
第四个四字节：最低，除以1000
第五个四字节：收盘，除以1000
第六个四字节：成交额，除以10
第七个四字节：成交量
第八个四字节：散户线
第九个四字节：似乎无用
第十个四字节：似乎与散户线有关

下边以601988为例说明一下：
打开文件601988.day，看到最后四十个字节为：

```
EC 1A 32 01 C6 0C 00 00
DA 0C 00 00 BC 0C 00 00 D0 0C 00 00 A9 5D 01 00
02 2C 04 00 D3 00 00 00 00 00 00 00 00 00 00 00
```

第一个四字节：EC 1A 32 01，十六进制为0x01321AEC，转换为十进制就是20060908，是日期
第二个四字节：C6 0C 00 00，十六进制为0x0CC6，转换为十进制就是3270，除以1000为3.27，是开盘价
第三个四字节：DA 0C 00 00，十六进制为0x0CDA，转换为十进制就是3290，除以1000为3.29，是最高价
第四个四字节：BC 0C 00 00，十六进制为0x0CBC，转换为十进制就是3260，除以1000为3.26，是最低价
第五个四字节：D0 0C 00 00，十六进制为0x0CD0，转换为十进制就是3280，除以1000为3.28，是收盘价
第六个四字节：A9 5D 01 00，十六进制为0x00015DA9，转换为十进制就是89513，除以10为8951.3，是成交额

第七个四字节：02 2C 04 00，十六进制为0x00042C02，转换为十进制就是273410，是成交量
第八个四字节：D3 00 00 00，十六进制为0xD3，转换为十进制为211，散户线

大智慧新一代逐笔数据格式(L2D文件格式)

在Level-2之前,沪深交易所提供的都是行情快照,大家看到的分笔成交其实是两次快照期间累计的成交量和最后一笔的价格,而逐笔成交则是真实的每笔成交价和成交量的明细数据。逐笔成交极大地提高了行情的透明度。

Level-2的逐笔成交数据是缓存在dzh2\data\sh\TEMP目录下，文件名以证券代码为名，后辍是.L2D，如果重启大智慧，L2D文件会全部被删除，而且临时文件只有在浏览过该证券的逐笔成交后才会生成。

此文档只公布部分L2D格式：

位置	字节	含义	示例	解码结果
0x28	4	分笔成交总笔数	7C 1C 00 00	
0x2a0	4	时间	5E E0 01 00	
0x2a0 + 4	4	价格	B9 17 00 00	
	2	数量		
	2	买卖标志		

20个字节为一个记录块，一共为320个记录块。

大智慧新一代4.01版本 最新行情数据结构

文件名是STKINFO60.DAT

文件的目录结构如下图：

代码10个字节

大智慧新一代一分钟线

大智慧1分钟线是通过分笔数据转换而来：

分笔数据解析格式得到结果如下:

1分钟线 由开盘 收盘 最高 最低 成交量 成交金额组成

举例说明 9:31一分钟线如何获得:

开盘	就是9:31的第一个时间的价格 15.35
收盘	15.40
最高	9:31此分钟最大值
最低	同上 求最小值
成交量	此一分钟现手和 666 + 815 + … + 3599
成交金额	每秒价格*现手 然后将结果相加。 15.35*666 + 15.38*815 + … 154.*3599

大智慧新一代分笔数据PRP格式(大智慧L2分笔数据PRP格式)

大智慧新一代分笔数据PRP格式(大智慧L2分笔数据PRP格式)

分笔数据保存 C:\dzh2\data\sh\20071025.prp

起止地址 数据内容 数据含义 数据类型
00 - 03 F4 9B 13 FC 日线文件标志 int
04 - 07 10 02 00 00 未知 int
08 - 0B 00 43 DF 46 1970.01.01 00:00:00始的秒数 int
0C - 0F 5D 05 00 00 证券总数 int
10 - 13 ED 20 00 00 未知 int
14 - 17 ED 20 00 00 未知 int
18 - 21 30 30 30 30 30 31 00 00 证券代码 byte[10]
22 - 25 9C 12 00 00 日分笔记录数 int
26 - 57 00 00 BA 03...FF FF 记录块号 char[25]

00041000 记录页起始点，

每记录长: 0x34 = 52
每页记录数: 0xEC = 236
每页长: 每记录长0x34 * 每页记录数0xEC = 0x2FF0 = 12272
地 址 数据内容 数据含义 数据类型
41000 - 41003 35 FA DF 46 1970.01.01 00:00:00 始的秒数 int
41004 - 41007 00 00 18 41 最新价 float
41008 - 4100B 00 80 B4 43 累计成交量 float
4100C - 4100F 80 46 A7 48 累计成交金额 float

41010 - 41011 51 9C 累计成交笔数 char
41012 - 41013 00 00 未知 char
41014 10 累计成交笔数的溢出标志（00|10） byte
41015 80 买入,卖出标识(80|E0买入，C0|A0卖出) byte

41016 - 41017 23 01 委买量1 char
41018 - 41019 8E 5B 委买量2 char
4101A - 4101B 80 27 委买量3 char
4101C - 4101D 8E 5B 委买量4 char
4101E - 4101F B8 40 委买量5 char

41020 - 41021 23 01 委卖量1 char
41022 - 41023 8E 5B 委卖量2 char
41024 - 41025 80 27 委卖量3 char
41026 - 41027 8E 5B 委卖量4 char
41028 - 41029 B8 40 委卖量5 char

4102A 16 委买价1 与成交价的差 byte
4102B 9A 委买价2 与成交价的差 byte
4102C 80 委买价3 与成交价的差 byte
4102D 40 委卖价4 与成交价的差 byte
4102E 30 委卖价5 与成交价的差 byte

4102F 57 委卖价1 与成交价的差 byte
41030 68 委卖价2 与成交价的差 byte
41031 69 委卖价3 与成交价的差 byte
41032 7A 委卖价4 与成交价的差 byte
41033 81 委卖价5 与成交价的差 byte

注意：
1.数据类型为JAVA数据类型定义，实际读数据时应该将数据十六进制取反，如：41000 - 41003的数据为 35 FA DF 46，实际应该读成：0x46DFFA35 = 1189083701， 这个1189083701值就是距1970.01.01 00:00:00 始的实际秒数。其他int,float,char都是这样取反的。

2.委买卖与成交价的差，是小数点后的整数差，如果买卖标识为80或C0，成交价小数点后是两位（股票），如果买卖标识为E0或A0，成交价小数点后是三位（权证等）

3.累计成交笔数如果溢出，则41014位上的值不为00，而是10或者20，30等，成交笔数计算方法应为：41014位上的值取第一位，10取前面的1，20取前面的2。。。 再与41010 - 41011位上的值合并成一个16进制串，本例中，实际值应该为0x19C51。

4.记录块号计算方法：char[0] = 0x00 第一个记录块起始地址为：41000h + 0x0000 * 0x2FF0 = 41000h，char[1] = 0x03BA 则第二个记录块起始地址为：41000h + 0x03BA * 0x2FF0 = 0x00077F78

5.记录块号计算方法：char[0] = 0x00 第一个记录块起始地址为：41000h + 0x0000 * 0xEC = 41000h，char[1] = 0x03BA 则第二个记录块起始地址为：41000h + 03BA * 0xEC = 0x00077F78
上面的页长应该是0x2FF0，而不是0xEC。

大智慧新一代Level-2日线文件day.dat数据格式

数据格式几乎与分析家相同

起止地址 数据内容 数据含义 数据类型
00 - 03 F4 9B 13 FC 文件标志 int
04 - 07 00 06 00 00 未知 int
08 - 0B 00 00 00 00 保留 int
0C - 0F 97 04 00 00 证券总数 int
10 - 13 00 18 00 00 未知 int

14 - 17 DB 17 00 00 未知 int
18 - 21 31 41 30 30 30...FF 证券代码 byte[10]
22 - 25 B0 09 00 00 日线记录数 int
26 - 57 00 00 25 04...FF FF 记录块号 char[25]
.....
记录块开始于0x41000
41000 - 41003 80 47 B2 2B 日期 int
41004 - 41007 B9 1E 25 41 开盘价 float
41008 - 4100B CD CC 4C 41 最高价 float
4100C - 4100F EC 51 18 41 最低价 float
41010 - 41013 9A 99 41 41 收盘价 float
41014 - 41017 80 06 B2 47 成交量 float
41018 - 4101B 40 1C BC 4C 成交金额 float
4101C - 4101D 00 00 上涨家数 char
4101E - 4101F 00 00 下跌家数 char

分析家数据格式

分析家目录结构

SUPERSTK下的文件:
SYS.DTA 存放系统提供的公式
BLOCK文件夹下的文件： *.IBK 板块指数定义 *.BLK 板块定义
BLOCK文件夹下的文件： *.IBK 板块指数定义 *.BLK 板块定义
*.EBK 条件选股结果
SELF 文件夹下的文件： *.WSV 保存页面文件 ALERT.DAT 历史预警纪录 EXTDATA.INF 扩展数据定义 *.CEP 保存组合条件选股条件
TEMPCMPD.CEP测试附加条件 *.INV 用户个人投资纪录 *.TPT 保存指标模板 SELF年月日.DTA 每日自动公式备份文件
TEST 文件夹下的文件： *.TST 存放系统测试结果 *.OPT 存放参数优化的结果
PARAM参数指引文件夹 *.PRM 存放参数指引的结果
TABLE文件夹下的文件：
*.ESS数据表文件
*.ESD数据表文件（带数据保存）
SelfData文件夹下的文件：
*.str 字符串数据
*.oth 与股票无关序列值数据
Pattern 文件夹下的文件
*.PIN 模式匹配设计
*.PWT模式匹配方法
SpotAna文件夹下的文件:
*.SPT 定位分析结果
Relate文件夹下的文件:
*.RTL 相关分析结果

Posible文件夹下的文件:
*.PSB 预测分布设计
DATA件夹下的文件： DAY.DAT 日线数据 EXTDAY.DAT 扩展数据 MIN.DAT 5分钟线数据 REPORT.DAT 当天的分笔成交数据 STKINFO.DAT 代码表/即时行情数据/财务数据/除权数据
NEWS文件夹下的文件：
*.PRP 历史回忆数据，一天一个文件 NEWS文件夹下的文件： *.TXT 财经报道、上交所公告、深交所公告

分析家3.1x版日线数据存储格式

上海日线存储文件为:\superstk\data\sh\day.dat
深圳日线存储文件为:\superstk\data\sz\day.dat

以上海日线存储文件day.dat为例：

```
00000h: F4 9B 13 FC 10 02 00 00-00 00 00 00 D1 04 00 00
00000h: 81 0C 00 00 48 0C 00 00-31 41 30 30 30 31 00 FF
00020h: FF FF B0 09 00 00 16 00-17 00 18 00 19 00 1A 00
00030h: 1B 00 1C 00 1D 00 1E 00-07 0A FF FF FF FF FF FF
00040h: FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
00050h: FF FF FF FF FF FF FF FF
......
......
41000h: 80 47 B2 2B B9 1E 25 41-CD CC 4C 41 EC 51 18 41
41010h: 9A 99 41 41 80 06 B2 47-40 1C BC 4C 00 00 00 00
41020h: 00 3C B6 2B 34 33 3F 41-AF 47 49 41 01 00 30 41
41030h: 34 33 3B 41 00 07 12 47-A4 3C 26 4C 00 00 00 00
```

该文件格式与磁盘文件物理存储方式类似：

起止地址 数据内容 数据含义 数据类型
00 - 03 F4 9B 13 FC 日线文件标志 Integer
04 - 07 10 02 00 00 保留 Integer
08 - 0B 00 00 00 00 保留 Integer
0C - 0F D1 04 00 00 证券总数 Integer
10 - 13 81 0C 00 00 需添加之起始块号 Integer
14 - 17 48 0C 00 00 当前最后空块号 Integer
18 - 21 31 41 30 30 30...FF 证券代码 Char[10]
22 - 25 B0 09 00 00 日线记录数 Integer
26 - 57 16 00 17 00...FF FF 记录块号 Word[25]
......
41000 - 41003 80 47 B2 2B 日期 Integer
41004 - 41007 B9 1E 25 41 开盘价 Single
41008 - 4100B CD CC 4C 41 最高价 Single
4100C - 4100F EC 51 18 41 最低价 Single
41010 - 41013 9A 99 41 41 收盘价 Single
41014 - 41017 80 06 B2 47 成交量 Single

41018 - 4101B 40 1C BC 4C 成交金额 Single
4101C - 4101D 00 00 上涨家数 Word
4101E - 4101F 00 00 下跌家数 Word

注：
1）起止地址、数据内容为十六进制，数据类型为 Delphi 下之定义。
2）从18h开始至40017h每64byte为一条股票数据分配记录，含义如上表18h - 57h所示；
3）从41000h开始每8KB为一股票数据存储块，每个股票数据存储块共存储256条日线记录，每一条记录的长度为32 byte（含义如上表；上涨家数及下跌家数只对指数有效）；
4）从41000h开始的8KB为第0号数据存储块，以后类推；
5）系统对每个股票日线数据存储以存储块为单位进行分配。
6）具体应用实例分析家数据管理程序。
7) 日期字段的意义为：实际日期 = StrToDate(^1970-01-01^)+(日期字段 div 86400)；即“日期字段”除以86400所得数为实际日期距1970年01月01日的天数。

分析家分笔数据格式

分析家的分笔数据文件(*.prp)格式:
0 - 03 F4 9B 13 FC 分笔数据文件标志 Integer
04 - 07 10 02 00 00 保留 Integer
08 - 0B 00 00 00 00 保留 Integer
0C - 0F D1 04 00 00 证券总数 Integer
10 - 13 81 0C 00 00 需添加之起始块号 Integer
14 - 17 48 0C 00 00 当前最后空块号 Integer
18 - 21 31 41 30 30 30...FF 证券代码 Char[10]
22 - 23 B0 09 日线记录数 Integer
24 - 25 FF FF 保留 Integer
26 - 57 16 00 17 00...FF FF 记录块号 Word[25]

----------------|
41000 - 41003 80 47 B2 2B 日期和时间 Long
|
41004 - 41007 B9 1E 25 41 最新价 Single
|
41008 - 4100B CD CC 4C 41 当天累计成交量 Single
|
4100C - 4100F EC 51 18 41 当天累计成交额 Single
|
41010 - 41015 9A 99 41 80 06 B2 买一量,买二量,买三量 integer
|每36个字节为一条分笔成交记录
41016 - 4101B 80 05 B2 70 12 B1 卖一量,卖二量,卖三量 integer |
4101B - 4101D CD CC 4C 买一,买二,买三 byte
|
4101E - 41020 31 41 30 卖一,卖二,卖三 byte
|
41021 - 41022 1A 2B 保留字节

|

1）从18h开始至40017h每64byte为一条股票数据分配记录，含义如上表18h - 57h所示；
2）从41000h开始每113*36=4068
byte为一个股票记录数据存储块，每个股票数据存储块共存储113条分笔数据记录，每一条记录的长度为36 byte
3) bid1=最新价 - 0.01 * 买一 依此类推 （股票）
ask1=最新价 + 0.01 * 卖一 依此类推

或 bid1=最新价 - 0.001 * 买一 依此类推 （基金和权证）
ask1=最新价 + 0.001 * 卖一 依此类推

pData->m_pDataEx.m_fSellPrice[0]=3900.000
并且财务数据pData->m_pfFinData[6]流通盘大小为0。

关于作分析家分笔成交数据存储格式和作股票分析

分析家分笔成交数据存储格式:

上海当天分笔成交数据存储文件为:\superstk\data\sh\report.dat
深圳当天分笔成交数据存储文件为:\superstk\data\sz\report.dat
20020801.prp,20020802.prp,.... 20020823.prp 等文件是历史分笔成交数据文件,
上海历史分笔成交数据存储文件目录为:\superstk\data\sh\
深圳历史分笔成交数据存储文件目录为:\superstk\data\sz\
起止地址 数据内容 数据含义 数据类型
00 - 03 F4 9B 13 FC 日线文件标志 Integer
04 - 07 10 02 00 00 保留 Integer
08 - 0B 00 91 40 3D 保留 Integer
0C - 0F D2 02 00 00 证券总数 Integer
10 - 13 51 05 00 00 需添加之起始块号 Integer
14 - 17 50 05 00 00 当前最后空块号 Integer
18 - 21 33 39 39 30 证券代码 Char[10]
22 - 25 30 31 00 FF 日线记录数 Integer
26 - 57 FF FF BA 03...FF FF 记录块号 Word[25]
.....
地 址 数据内容 数据含义 数据类型
41000 - 41003 98 00 66 3D 日期 Integer
41004 - 41007 00 00 18 41 最新价 Single
41008 - 4100B 00 80 B4 43 累计成交量 Single
4100C - 4100F 80 46 A7 48 累计成交金额 Single
41010 - 41011 EA AA 委买量1 Integer
41012 - 41013 2A 3F 委买量2 Integer
41014 - 41015 24 57 委买量3 Integer
41015 - 41017 23 01 委卖量1 Integer
41018 - 41019 8E 5B 委卖量2 Integer
4101A - 3101B 80 40 委卖量3 Integer

4101C 16 委买价1的小数部分 Byte
4101D 9A 委买价2的小数部分 Byte
4101E 80 委买价3的小数部分 Byte
4101F 40 委卖价1的小数部分 Byte
41020 30 委卖价2的小数部分 Byte
41021 57 委卖价3的小数部分 Byte
41022 - 41023 00 80 买入,卖出标识 Byte
1.)从18h开始至40017h每64byte为一条股票数据分配记录，含义如上表18h - 57h所示；
2.)从41000h开始每4068byte为一股票数据存储块，每个股票数据存储块共存储113条记录，每一条记录的长度为36 byte:
具体含义如上表41000h - 41023h所示；
3.) 日期字段意义均为：实际日期 = CDate('1970-01-01')+(日期字段 div 86400)；即“日期字段”除以86400所得数为
实际日期距1970年01月01日的天数。
4.)委买,委卖价由最新价加委买,委卖价小数部分得到,如:
委买价1的小数部分=16h(<80h), 则委买价1=最新价+32/100; (16h=32d)
委买价2的小数部分=9Ah(>80h), 则委买价2=最新价-(256-154)/100; (9Ah=154d)
5.)单笔成交量用本笔累计数减上笔累计数得到,
若买入,卖出标识为80h,则单笔成交量是买入量;
若买入,卖出标识为C0h,则单笔成交量是卖出量。

分析家的财经数据文件的格式（指的是安装数据*.fin）

下面是VB编写的格式:
每纪录为476个字节
排序方式与有效日线一致
文件后段为代码、名称、拼音简码及财务数据

```
Type 分析家除权记录
日期 As Long
送股 As Single
配股 As Single
配价 As Single
派息 As Single
End Type

Type 分析家除权
标志 As Long
记录数 As Long
股票数 As Long
'no As Integer
除权数 As Integer
无配股 As Boolean
记录1 As 分析家除权记录
记录2 As 分析家除权记录
记录3 As 分析家除权记录
记录4 As 分析家除权记录
记录5 As 分析家除权记录
```

记录6 As 分析家除权记录
记录7 As 分析家除权记录
记录8 As 分析家除权记录
记录9 As 分析家除权记录
记录10 As 分析家除权记录
记录11 As 分析家除权记录
记录12 As 分析家除权记录
记录13 As 分析家除权记录
记录14 As 分析家除权记录
记录15 As 分析家除权记录
记录16 As 分析家除权记录
记录17 As 分析家除权记录
记录18 As 分析家除权记录
记录19 As 分析家除权记录
记录20 As 分析家除权记录
记录21 As 分析家除权记录
记录22 As 分析家除权记录
记录23 As 分析家除权记录
End Type

分析家安装数据格式

dad数据格式

前20个字节为头信息：

1~4? ? ? ? 为安装数据的标识(33 FC 19 8C)
5~8? ? ? ? 为 ?? ?? ?? ?? 未知
9~12? ? ? ? 为本文件的股票数
13~16? ? ? ? 为00 00 00 00
17~20? ? ? ? 为FF FF FF FF

对于单日的安装数据

标识(33 FC 19 8C) ?? ?? ?? ?? 本文件的股票数 00 00 00 00
FF FF FF FF SHXX(SZXX) XXXX(XX 00 00) 00(00 00 00)|(重复上一只股票的最低价的后三位，不指逻辑上的，如果是第一只则用40 00 00)
?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??
日期 开盘价 最高价 最低价
收盘价 成交量(手) 成交额(元) ?? ?? ?? ??
FF FF FF FF 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

对于多日的安装数据

标识(33 FC 19 8C) ?? ?? ?? ?? 本文件的股票数 00 00 00 00
FF FF FF FF SHXX(SZXX) XXXX(XX 00 00) 00(00 00 00)|(重复上一只股票的最低价的后三位，不

指逻辑上的，如果是第一只则用40 00 00)
?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??
日期 开盘价 最高价 最低价
收盘价 成交量(手) 成交额(元) ?? ?? ?? ??
日期 开盘价 最高价 最低价
收盘价 成交量(手) 成交额(元) ?? ?? ?? ??
日期 开盘价 最高价 最低价
收盘价 成交量(手) 成交额(元) ?? ?? ?? ??
日期 开盘价 最高价 最低价
收盘价 成交量(手) 成交额(元) ?? ?? ?? ??
日期 开盘价 最高价 最低价
收盘价 成交量(手) 成交额(元) ?? ?? ?? ??
FF FF FF FF 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

每个日K线为36字节

time_t? ? ? ? Date;? ? ? ? //日期,UCT方式
float? ? ? ? OPen;? ? ? ? //开盘(元)
float? ? ? ? High;? ? ? ? //最高价(元)
float? ? ? ? Low;? ? ? ? //最低价(元)
float? ? ? ? Close;? ? ? ? //收盘(元)
float? ? ? ? Money;? ? ? ? //成交量(手)
float? ? ? ? Volume;? ? ? ? //成交额(元)
float? ? ? ? Nouse1;? ? ? ? //未用

恒生数据数据结构

(一)恒生日线文件数据结构

沪市日线文件路径：默认在HSNEW\DATA\SHASE\DAY下。
深市日线文件路径：默认在HSNEW\DATA\SZNSE\DAY下。

日线文件命名规则：股票代码.DAY

每个日K线为40字节，具体如下：
Date:LongInt; //日期
OPen:LongInt; //开盘(元/1000)
High:LongInt; //最高价(元/1000)
Low:LongInt; //最低价(元/1000)
Close:LongInt; //收盘(元/1000)
Money:LongInt; //成交额(千元)
Volume:LongInt; //成交量(手)
Nouse1:LongInt; //没用
Nouse2:LongInt; //没用
Nouse3:LongInt; //没用

钱龙数据格式

钱龙代码表文件格式

沪市代码表文件路径：默认为ML30\DATA\SHASE\NAMETBL.SHA
深市代码表文件路径：默认为ML30\DATA\SZNSE\NAMETBL.SZN

```
First0:ShortInt; //1字节，只能为00
Name:array [1..8] of Char;
Kind:ShortInt;
Code:array [1..6] of Char;//代码深圳为XXXX__
YesClose:LongInt;//昨收
Open:LongInt;//开盘
High:LongInt;//最高
Low:LongInt;//最低
Close:LongInt;//收盘
Volume:LongInt;//总手
Money:LongInt;//金额
Buy1M:LongInt;//买一价
Buy1V:LongInt;//买一量
Buy2M:LongInt;//买二价
Buy2V:LongInt;//买二量
Buy3M:LongInt;//买三价
Buy3V:LongInt;//买三量
Sale1M:LongInt;//卖一价
Sale1V:LongInt;//卖一量
Sale2M:LongInt;//卖二价
Sale2V:LongInt;//卖二量
Sale3M:LongInt;//卖三价
Sale3V:LongInt;//卖三量
Per:SmallInt;//只能为100？？
NowV:LongInt;//现手
```

钱龙网络版动态数据格式（即ml30\lond\dat下的数据格式）

```
// Set the default value
#define EXCH_MIN 240
#define EXCH_A 9*60+30
#define EXCH_AE 11*60+30
#define EXCH_B 13*60+00
```

```
#define EXCH_BE 15*60+00
#define SEP_TAG 0xffffffff
#define TRA_MAXN 1000
#define INFO_PARA 0x84
#define TYPE_NUM 0x4
#define FALSE 0x0
#define TRUE 0x1
#define SLHS_SIZE 0x300
#define SH_ZS ^a^
#define SH_AG ^b^
#define SH_BG ^c^
#define SH_ZQ ^d^
#define SZ_ZS ^A^
#define SZ_AG ^B^
#define SZ_BG ^C^
#define SZ_ZQ ^D^
#define DH_ZS 0x10
#define DH_AG 0x11
#define DH_BG 0x12
#define DH_ZQ 0x13
#define DZ_ZS 0x20
#define DZ_AG 0x21
#define DZ_BG 0x22
#define DZ_ZQ 0x23

struct APD_stock_data{ // in 199YMMDD.DAT
unsigned char data_id; // =0x73 with min_data,// =0x09 without min_data,// =0x20
is deleted
unsigned char stock_type; // many be a,b,c,d or A,B,C,D
unsigned char stock_code[6]; // stock code
unsigned char stock_name[8]; // name of the stock
unsigned long last_close_price;
unsigned long open_price;
unsigned long high_price;
unsigned long low_price;
unsigned long close_price;
unsigned long total_volume;
unsigned long total_value;
};

struct APD_stock_min{ // store data for normal shares
unsigned long cur_price; // price of every minutes
unsigned long total_volume; // summary volume by this minute
unsigned int average_percent; // swing percent of the average value<
};

struct APD_index_min{ // store data for index
unsigned long cur_price; // index of every minutes
unsigned long total_volume; // summary volume by this minute
unsigned int average_percent; // swing percent of the average index
unsigned int buy_vigour; // the vigour value of buying
unsigned int sell_vigour; // the vigour value of selling
};

// Develop data files by appdata structured upwards
```

```
struct Data_5min{ // in .nmn files
unsigned long min_off; // Format is MMDDHHMM
unsigned long open_price; // 0.001
unsigned long high_price; // 0.001
unsigned long low_price; // 0.001
unsigned long close_price; // 0.001
unsigned long min_amount; // 100
unsigned long min_volume; // 100
unsigned long time_count; // sum trade time
unsigned char share_value; // share value
unsigned char share_number; // share break number
unsigned int share_bonus; // share bonus
unsigned long shares_number; // sum number
};

struct His_data{ // in lonhis.???
unsigned long cur_price;
unsigned long total_volume; // total volume
unsigned long total_value; // total value by this minute
unsigned long buy_volume; // initiactive total buy volume
unsigned long sell_volume; // initiactive total sell volume
unsigned int rise_num; // summary rise shares of this minute
unsigned int fall_num; // summary fall shares of this minute
unsigned int average_percent; // swing percent of the average index
unsigned int buy_vigour; // the vigour value of buying
unsigned int sell_vigour; // the vigour value of selling
unsigned int value_ADL; // the value of current ADL index
unsigned int swing_flag; // the flag of rise or fall
};

struct Tra_data{ // in lontra.???
unsigned int time_off;
unsigned long cur_price;
unsigned long total_vol;
unsigned long buy_price;
unsigned long sell_price;
};

struct Info_data{ // in loninfo.???
unsigned char stock_id; // This byte is to id stock
unsigned char data_tag; // always be 0 to id
unsigned int stock_num;
unsigned int start_num;
unsigned int max_tra;
unsigned int min_exch;
unsigned int exch_min;
unsigned int A_begin_time;
unsigned int A_end_time;
unsigned int B_begin_time;
unsigned int B_end_time;
};

struct Cdp_data{ // shacdp.dat of HXTW
unsigned char stock_code[6];
```

```
unsigned char reserved[20];
unsigned long reserved2;
};

// some dynamic data files structured upwards

struct Data_day{ // in .day files
unsigned long day_date; // Format is YYYYMMDD
unsigned long open_price; // 0.001
unsigned long high_price; // 0.001
unsigned long low_price; // 0.001
unsigned long close_price; // 0.001
unsigned long day_amount; // 1000
unsigned long day_volume; // 100
unsigned long time_count; // sum trade time
unsigned char share_value; // share value
unsigned char share_number; // share break number
unsigned int share_bonus; // share bonus
unsigned long shares_number; // sum number
};

struct Slon_para{ // in file slonpara.dat
unsigned int sh_A_para; // Offset of SH_A stock
unsigned int sh_B_para;
unsigned int sh_C_para;
unsigned int sh_total_para;
unsigned int sz_A_para;
unsigned int sz_B_para;
unsigned int sz_C_para;
unsigned int sz_total_para;
unsigned int data_tag[3];
};

struct Slon_day{ // in .day files of Slon
unsigned long day_date; // Format is YYYYMMDD
unsigned long open_price; // 0.001
unsigned long close_price; // 0.001
unsigned long high_price; // 0.001
unsigned long low_price; // 0.001
unsigned long day_amount; // 1000
unsigned long day_volume; // 100
unsigned int reserved[6]; // Some infomation I don^t know
};

struct Slon_HS{ // in lonhs.dat of Slon
unsigned int data_tag1; // Some infomation I don^t know
unsigned char stock_type; // =0xff is delete, =0x64 is OK
unsigned long stock_code; // number of stock code
unsigned char stock_name[8]; // name of stock
unsigned char data_tag2; // =0x0, Some infomation I don^t know
unsigned int data_tag3; // =0x64, Some infomation I don^t know
unsigned int last_close_price;
unsigned int open_price;
unsigned long PMA5_volume; // The volume of 5day average
unsigned int high_price;
```

```
unsigned int low_price;
unsigned int close_price;
unsigned long data_tag4; // Some infomation I don^t know
unsigned long total_volume1; // Total volume
unsigned long total_volume2; // Total volume
unsigned int buy_1_price; // The price of buying now
unsigned char buy_2_sub; // The offset of buy 1 and buy 2 price
unsigned char buy_3_sub; // The offset of buy 1 and buy 3 price
unsigned long buy_1_volume;
unsigned long buy_2_volume;
unsigned long buy_3_volume;
unsigned int sell_1_price; // The price of selling now
unsigned char sell_2_add; // The offset of sell 1 and sell 2 price
unsigned char sell_3_add; // The offset of sell 1 and sell 3 price
unsigned long sell_1_volume;
unsigned long sell_2_volume;
unsigned long sell_3_volume;
unsigned int refx;
unsigned int refy;
unsigned int hisfptr;
unsigned int numppg;
unsigned char pgnum;
unsigned int hisptr00;
unsigned int hisptr01;
unsigned int hisptr02;
unsigned int hisptr03;
unsigned int hisptr04;
unsigned int hisptr05;
unsigned int hisptr06;
unsigned int hisptr07;
unsigned int hisptr08;
unsigned int hisptr09;
unsigned int data_tag5;
unsigned long cur_volume;
unsigned char reserved1[12];
unsigned int minfptr;
unsigned int minsize;
unsigned long data_tag6;
unsigned long out_volume;
unsigned char reserved2[133];
unsigned char data_tag7; // always 07
unsigned char reserved3[145];
unsigned char end_hour;
unsigned char end_minute;
unsigned int lastmin_vol;
unsigned int data_tag8;
unsigned long total_volume3;
unsigned int cur_price;
unsigned long average_price;
unsigned char reserved4[82];
};

/* SLON data structure from MR.ZHAO BIAO

Lonhs.dat结构分析（QB）
```

位置	长度	类型	说明
1	2	INTEGER	**********
2	1	STRING	标志，FF->delete 64->ok
3	4	LONG	股票代码
4	8	STRING	股票名称
5	1	STRING	00-> ? **********
6	2	INTEGER	64-> ? **********
7	2	INTEGER	前收盘价
8	2	INTEGER	开盘价
9	4	LONG	五日均量
10	2	INTEGER	最高价
11	2	INTEGER	最低价
12	2	INTEGER	收盘价

钱龙3.0版数据存储格式

上海日线存储路径为:\ml30\data\shase\day,文件扩展名为:.day
上海周线存储路径为:\ml30\data\shase\week,文件扩展名为: .wek
上海月线存储路径为:\ml30\data\shase\month,文件扩展名为: .mnt
深圳日线存储路径为:\ml30\data\sznse\day
深圳周线存储路径为:\ml30\data\sznse\week
深圳月线存储路径为:\ml30\data\sznse\month

以深发展日线为例:
1A76:0100 D6 CD 2F 01 52 07 01 00-52 07 01 00 52 07 01 00
1A76:0110 52 07 01 00 86 0F 00 00-4D 02 00 00 00 00 00 00
1A76:0120 00 00 00 00 00 00 00 00-D7 CD 2F 01 60 03 01 00
1A76:0130 60 03 01 00 60 03 01 00-60 03 01 00 82 05 00 00
1A76:0140 D4 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00

每一条记录的长度为40字节:
1-4字节为日期,D6 CD 2F 01转换为十进制是:19910102
5-8字节为开盘价*1000
9-12字节为最高价*1000
13-16字节为最低价*1000
17-20字节为收盘价*1000
21-24字节为成交量(手)
25-28字节为成交金额
其余12字节未使用

另:周线,月线格式与日线格式一致.

下面是我用C语言编的一个显示深发展日线的小程序,运行时要将000001.day拷到当前目录.

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
```

```
typedef struct {
unsigned long date;
unsigned long open;
unsigned long high;
unsigned long low;
unsigned long close;
unsigned long travl;
unsigned long traca;
char unuse[12];
} RECORD;

RECORD reco;
int readrec(FILE *);

void main()
{
FILE *fp;
if((fp = fopen("000001.day","rb")) == NULL) // 打开深发展日线
{ printf("Error: Can^t open 000001.DAY !\n");
exit(0); }
readrec(fp);
fclose(fp);
if(getch()==0) getch();
exit(0);
}

int readrec(FILE *fp)
{
float fn;
while (! feof(fp)) {
fread(&reco,sizeof(RECORD),1,fp);
printf("%10lu ",reco.date);
fn=float(reco.open)/1000;
printf("%8.2f ",fn);
fn=float(reco.high)/1000;
printf("%8.2f ",fn);
fn=float(reco.low)/1000;
printf("%8.2f ",fn);
fn=float(reco.close)/1000;
printf("%8.2f ",fn);
printf("%8lu ",reco.travl);
printf("%8lu\n",reco.traca);
}
printf("\n");
return 0;
}
```

钱龙数据格式c++语言定义版本

钱龙数据格式c++语言定义版本

```
//      钱龙网络版动态数据格式（即ml30\lond\dat下的数据格式）
//      Set the default value
#define EXCH_MIN 240
```

```
#define EXCH_A       9*60+30
#define EXCH_AE     11*60+30
#define EXCH_B       13*60+00
#define EXCH_BE     15*60+00
#define SEP_TAG      0xffffffff
#define TRA_MAXN          1000
#define INFO_PARA 0x84
#define TYPE_NUM 0x4
#define SLHS_SIZE   0x300
#define SH_ZS 'a'
#define SH_AG        'b'
#define SH_BG        'c'
#define SH_ZQ         'd'
#define SZ_ZS  'A'
#define SZ_AG 'B'
#define SZ_BG 'C'
#define SZ_ZQ 'D'
#define DH_ZS         0x10
#define DH_AG        0x11
#define DH_BG        0x12
#define DH_ZQ         0x13
#define DZ_ZS 0x20
#define DZ_AG        0x21
#define DZ_BG        0x22
#define DZ_ZQ        0x23

#pragma  pack(1)

struct QL_APD_stock_data{    // in 199YMMDD.DAT
BYTE data_id; // =0x73 with min_data ,
// =0x09 without min_data ,
// =0x20 is deleted

BYTE stock_type; // many be a,b,c,d or A,B,C,D
BYTE stock_code[6]; // stock code
BYTE stock_name[8]; // name of the stock

DWORD last_close_price;
DWORD open_price;
DWORD high_price;
DWORD low_price;
DWORD close_price;
DWORD total_volume;
DWORD total_value;
};

struct QL_APD_stock_min{ // store data for normal shares
DWORD cur_price; // price of every minutes
DWORD total_volume; // summary volume by this minute
WORD average_percent; // swing percent of the average value
};

struct QL_APD_index_min{ // store data for index
DWORD cur_price; // index of every minutes
DWORD total_volume; // summary volume by this minute
WORD average_percent; // swing percent of the average index

WORD buy_vigour; // the vigour value of buying
WORD sell_vigour; // the vigour value of selling
```

```
};

// Develop data files by appdata structured upwards

struct QL_Data_5min{ // in .nmn files
DWORD min_off; // Format is MMDDHHMM
DWORD open_price; // 0.001
DWORD high_price; // 0.001
DWORD low_price; // 0.001
DWORD close_price; // 0.001

DWORD min_amount; // 100
DWORD min_volume; // 100

DWORD time_count; // sum trade time
BYTE share_value; // share value
BYTE share_number; // share break number
WORD share_bonus; // share bonus
DWORD shares_number; // sum number
};

struct QL_His_data{ // in lonhis.???
DWORD cur_price;
DWORD total_volume; // total volume
DWORD total_value; // total value by this minute
DWORD buy_volume; // initiactive total buy volume
DWORD sell_volume; // initiactive total sell volume
WORD rise_num; // summary rise shares of this minute
WORD fall_num; // summary fall shares of this minute

WORD average_percent; // swing percent of the average index
WORD buy_vigour; // the vigour value of buying
WORD sell_vigour; // the vigour value of selling
WORD value_ADL; // the value of current ADL index
WORD swing_flag; // the flag of rise or fall
};

struct QL_Tra_data{ // in lontra.???
WORD time_off;
DWORD cur_price;
DWORD total_vol;
DWORD buy_price;
DWORD sell_price;
};

struct QL_Info_data{ // in loninfo.???
BYTE stock_id; // This byte is to id stock
BYTE data_tag; // always be 0 to id
WORD stock_num;
WORD start_num;
WORD max_tra;
WORD min_exch;

WORD exch_min;
WORD A_begin_time;
WORD A_end_time;
WORD B_begin_time;
WORD B_end_time;
};
```

```
struct QL_Cdp_data{ // shacdp.dat of HXTW
BYTE stock_code[6];
BYTE reserved[20];
DWORD reserved2;
};

// some dynamic data files structured upwards
struct QL_Data_day{       // in .day files
DWORD   day_date;         //日期    Format is XXMMDDHHMM for 5min, Format is YYYYMMDD for day
DWORD   open_price;       // 开盘 0.001
DWORD   high_price;       // 最高价 0.001
DWORD   low_price;        // 最低价 0.001
DWORD   close_price;      // 收盘 0.001
DWORD   day_amount;   // 成交额(千元) 1000
DWORD   day_volume;     // 成交量(手) 100

DWORD   time_count;     // sum trade time
BYTE        share_value;    // share value
BYTE        share_number;       // share break number
WORD      share_bonus;  // share bonus
DWORD   shares_number;      // sum number
};

struct QL_Stock_info_V302{ // in lonnow.??? of QL302S
BYTE data_id; // =0 is OK , =0xff is deleted
BYTE stock_name[8];
BYTE stock_type;
BYTE stock_code[6];
DWORD last_close_price;
DWORD open_price;
DWORD high_price;
DWORD low_price;
DWORD close_price;

DWORD total_volume;
DWORD total_value;

DWORD buy_1_price;
DWORD buy_1_volume;
DWORD buy_2_price;
DWORD buy_2_volume;
DWORD buy_3_price;
DWORD buy_3_volume;
DWORD sell_1_price;
DWORD sell_1_volume;
DWORD sell_2_price;
DWORD sell_2_volume;
DWORD sell_3_price;
DWORD sell_3_volume;

WORD reserved; // Most time is 0x0064
};

struct QL_Stock_info2_V304{ // in lonnow.??? of QL304S
BYTE data_id; // =0 is OK , =0xff is deleted
BYTE stock_name[8];
BYTE stock_type;
```

```
BYTE stock_code[6];
DWORD last_close_price;
DWORD open_price;
DWORD high_price;
DWORD low_price;
DWORD close_price;

DWORD total_volume;
DWORD total_value;

DWORD buy_1_price;
DWORD buy_1_volume;
DWORD buy_2_price;
DWORD buy_2_volume;
DWORD buy_3_price;
DWORD buy_3_volume;

DWORD sell_1_price;
DWORD sell_1_volume;
DWORD sell_2_price;
DWORD sell_2_volume;
DWORD sell_3_price;
DWORD sell_3_volume;

WORD reserved; // Most time is 0x0064
DWORD reserved2; // Maybe the PINYIN
};

struct QL_Sse_data{ // in sse20a.dat,sse21a.dat,sse22a.dat
BYTE data_id; // 0x30,0x31,0x32
BYTE stock_code[6];
BYTE stock_name[8];
BYTE stock_type1; // 0x30 or 0x31
BYTE stock_type2; // 00 or 01
BYTE reserved; // =0
};

struct QL_Name_table{ // in nametbl.*
BYTE data_id; // 0x10...
BYTE stock_code[6];
BYTE stock_name[8];
BYTE data_tag;
};
```

钱龙网络版动态数据格式（即ml30\lond\dat下的数据格式）

```
// Set the default value
#define EXCH_MIN 240
#define EXCH_A 9*60+30
#define EXCH_AE 11*60+30
#define EXCH_B 13*60+00
#define EXCH_BE 15*60+00
#define SEP_TAG 0xffffffff
```

```
#define TRA_MAXN 1000
#define INFO_PARA 0x84
#define TYPE_NUM 0x4
#define FALSE 0x0
#define TRUE 0x1
#define SLHS_SIZE 0x300
#define SH_ZS ^a^
#define SH_AG ^b^
#define SH_BG ^c^
#define SH_ZQ ^d^
#define SZ_ZS ^A^
#define SZ_AG ^B^
#define SZ_BG ^C^
#define SZ_ZQ ^D^
#define DH_ZS 0x10
#define DH_AG 0x11
#define DH_BG 0x12
#define DH_ZQ 0x13
#define DZ_ZS 0x20
#define DZ_AG 0x21
#define DZ_BG 0x22
#define DZ_ZQ 0x23

struct APD_stock_data{ // in 199YMMDD.DAT
unsigned char data_id; // =0x73 with min_data,// =0x09 without min_data,// =0x20 is deleted
unsigned char stock_type; // many be a,b,c,d or A,B,C,D
unsigned char stock_code[6]; // stock code
unsigned char stock_name[8]; // name of the stock
unsigned long last_close_price;
unsigned long open_price;
unsigned long high_price;
unsigned long low_price;
unsigned long close_price;
unsigned long total_volume;
unsigned long total_value;
};

struct APD_stock_min{ // store data for normal shares
unsigned long cur_price; // price of every minutes
unsigned long total_volume; // summary volume by this minute
unsigned int average_percent; // swing percent of the average value<
};

struct APD_index_min{ // store data for index
unsigned long cur_price; // index of every minutes
unsigned long total_volume; // summary volume by this minute
unsigned int average_percent; // swing percent of the average index
unsigned int buy_vigour; // the vigour value of buying
unsigned int sell_vigour; // the vigour value of selling
};

// Develop data files by appdata structured upwards

struct Data_5min{ // in .nmn files
unsigned long min_off; // Format is MMDDHHMM
unsigned long open_price; // 0.001
unsigned long high_price; // 0.001
unsigned long low_price; // 0.001
unsigned long close_price; // 0.001
```

```
unsigned long min_amount; // 100
unsigned long min_volume; // 100
unsigned long time_count; // sum trade time
unsigned char share_value; // share value
unsigned char share_number; // share break number
unsigned int share_bonus; // share bonus
unsigned long shares_number; // sum number
};

struct His_data{ // in lonhis.???
unsigned long cur_price;
unsigned long total_volume; // total volume
unsigned long total_value; // total value by this minute
unsigned long buy_volume; // initiactive total buy volume
unsigned long sell_volume; // initiactive total sell volume
unsigned int rise_num; // summary rise shares of this minute
unsigned int fall_num; // summary fall shares of this minute
unsigned int average_percent; // swing percent of the average index
unsigned int buy_vigour; // the vigour value of buying
unsigned int sell_vigour; // the vigour value of selling
unsigned int value_ADL; // the value of current ADL index
unsigned int swing_flag; // the flag of rise or fall
};

struct Tra_data{ // in lontra.???
unsigned int time_off;
unsigned long cur_price;
unsigned long total_vol;
unsigned long buy_price;
unsigned long sell_price;
};

struct Info_data{ // in loninfo.???
unsigned char stock_id; // This byte is to id stock
unsigned char data_tag; // always be 0 to id
unsigned int stock_num;
unsigned int start_num;
unsigned int max_tra;
unsigned int min_exch;
unsigned int exch_min;
unsigned int A_begin_time;
unsigned int A_end_time;
unsigned int B_begin_time;
unsigned int B_end_time;
};

struct Cdp_data{ // shacdp.dat of HXTW
unsigned char stock_code[6];
unsigned char reserved[20];
unsigned long reserved2;
};

// some dynamic data files structured upwards

struct Data_day{ // in .day files
unsigned long day_date; // Format is YYYYMMDD
unsigned long open_price; // 0.001
unsigned long high_price; // 0.001
unsigned long low_price; // 0.001
```

```
unsigned long close_price; // 0.001
unsigned long day_amount; // 1000
unsigned long day_volume; // 100
unsigned long time_count; // sum trade time
unsigned char share_value; // share value
unsigned char share_number; // share break number
unsigned int share_bonus; // share bonus
unsigned long shares_number; // sum number
};

struct Slon_para{ // in file slonpara.dat
unsigned int sh_A_para; // Offset of SH_A stock
unsigned int sh_B_para;
unsigned int sh_C_para;
unsigned int sh_total_para;
unsigned int sz_A_para;
unsigned int sz_B_para;
unsigned int sz_C_para;
unsigned int sz_total_para;
unsigned int data_tag[3];
};

struct Slon_day{ // in .day files of Slon
unsigned long day_date; // Format is YYYYMMDD
unsigned long open_price; // 0.001
unsigned long close_price; // 0.001
unsigned long high_price; // 0.001
unsigned long low_price; // 0.001
unsigned long day_amount; // 1000
unsigned long day_volume; // 100
unsigned int reserved[6]; // Some infomation I don't know
};

struct Slon_HS{ // in lonhs.dat of Slon
unsigned int data_tag1; // Some infomation I don't know
unsigned char stock_type; // =0xff is delete, =0x64 is OK
unsigned long stock_code; // number of stock code
unsigned char stock_name[8]; // name of stock
unsigned char data_tag2; // =0x0, Some infomation I don't know
unsigned int data_tag3; // =0x64, Some infomation I don't know
unsigned int last_close_price;
unsigned int open_price;
unsigned long PMA5_volume; // The volume of 5day average
unsigned int high_price;
unsigned int low_price;
unsigned int close_price;
unsigned long data_tag4; // Some infomation I don't know
unsigned long total_volume1; // Total volume
unsigned long total_volume2; // Total volume
unsigned int buy_1_price; // The price of buying now
unsigned char buy_2_sub; // The offset of buy 1 and buy 2 price
unsigned char buy_3_sub; // The offset of buy 1 and buy 3 price
unsigned long buy_1_volume;
unsigned long buy_2_volume;
unsigned long buy_3_volume;
unsigned int sell_1_price; // The price of selling now
unsigned char sell_2_add; // The offset of sell 1 and sell 2 price
unsigned char sell_3_add; // The offset of sell 1 and sell 3 price
unsigned long sell_1_volume;
```

```
unsigned long sell_2_volume;
unsigned long sell_3_volume;
unsigned int refx;
unsigned int refy;
unsigned int hisfptr;
unsigned int numppg;
unsigned char pgnum;
unsigned int hisptr00;
unsigned int hisptr01;
unsigned int hisptr02;
unsigned int hisptr03;
unsigned int hisptr04;
unsigned int hisptr05;
unsigned int hisptr06;
unsigned int hisptr07;
unsigned int hisptr08;
unsigned int hisptr09;
unsigned int data_tag5;
unsigned long cur_volume;
unsigned char reserved1[12];
unsigned int minfptr;
unsigned int minsize;
unsigned long data_tag6;
unsigned long out_volume;
unsigned char reserved2[133];
unsigned char data_tag7; // always 07
unsigned char reserved3[145];
unsigned char end_hour;
unsigned char end_minute;
unsigned int lastmin_vol;
unsigned int data_tag8;
unsigned long total_volume3;
unsigned int cur_price;
unsigned long average_price;
unsigned char reserved4[82];
};

/* SLON data structure from MR.ZHAO BIAO
```

Lonhs.dat结构分析（QB）

位置	长度	类型	说明
1	2	INTEGER	**********
2	1	STRING	标志，FF->delete 64->ok
3	4	LONG	股票代码
4	8	STRING	股票名称
5	1	STRING	00-> ? **********
6	2	INTEGER	64-> ? **********
7	2	INTEGER	前收盘价
8	2	INTEGER	开盘价
9	4	LONG	五日均量
10	2	INTEGER	最高价
11	2	INTEGER	最低价
12	2	INTEGER	收盘价

钱龙权息数据WGT的数据格式股票数据

数据所在目录：C:\Program Files\qianlong\qijian\QLDATA\history\SHASE\weight

```
Public Type QLQX '权息数据WGT的数据格式
    Rq  As  Long    '日期-是一个21位(bit)的数,占用4个字节(32位),前12位表示年,接着的4位表示月,接着的5位表示日,剩下的位未使用。
  Sgs As Long '送股数- /10000=每10股送股数
  Pgs As Long '配股数- /10000=每10股配股数
  Pgj As Long '配股价- /1000
  HL As Long '红利  - /1000
  Zzs As Long '转增数- /10000
  Zgb As Long '总股本- 单位是万股
  LTG As Long '流通股- 单位是万股
  Memo As Long '备注
End Type

/*钱龙金典版除权(权息)数据*/
typedef struct QL_POWER_Tag
{
  DWORD   m_date;          // 日期 一个32位的无符号数，Format is MMDDHHMM
                    // 32位中前12位(int)是年，接下来的4位、5位分别是月、日，最后还有13位可以用来存时、分
  float   m_fM10GGive;     // 送股数- /10000=每10股送股数
  float   m_fM10GPei;      // 送股数- /10000=每10股送股数
  float   m_fPeiPrice;     // 配股价- /1000
  float   m_fProfit;       // 红利  - /1000
  float   m_fZzs;          // 转增数- /10000
  float   m_fZgb;          // 总股本- 单位是万股
  float   m_fLtg;          // 流通股- 单位是万股
  BYTE    m_ucMemo;        // 备注
}QL_POWER_t;
```

胜龙的数据格式

上海日线存储路径为:\slon\data\sh\day,文件扩展名为:.day

上海周线存储路径为:\slon\data\sh\week,文件扩展名为: .wek

上海月线存储路径为:\slon\data\sh\month,文件扩展名为: .mnt

深圳日线存储路径为:\slon\data\sz\day

深圳周线存储路径为:\slon\data\sz\week

深圳月线存储路径为:\slon\data\sz\month

以深发展日线为例:

0F74:0100 29 32 D0 FE AD F8 FE FF-AD F8 FE FF AD F8 FE FF

0F74:0110 AD F8 FE FF 77 F0 FF FF-B2 FD FF FF FF FF FF FF

0F74:0120 FE FF FF FF 94 E3 FB FF-28 32 D0 FE 9F FC FE FF

0F74:0130 9F FC FE FF 9F FC FE FF-9F FC FE FF 7D FA FF FF

0F74:0140 2B FF FF FF FF FF FF FF-FE FF FF FF 94 E3 FB FF

每一条记录的长度为40字节:

1-4字节为日期(如19910102)转换为整数的反码

5-8字节为开盘价*1000 的反码

9-12字节为最高价*1000 的反码

13-16字节为最低价*1000 的反码

17-20字节为收盘价*1000 的反码

21-24字节为成交量(手) 的反码

25-28字节为成交金额（万元）的反码

29-36字节未使用

37-40字节为除权价*1000 的反码

另:周线,月线格式与日线格式一致.

下面是我用C语言编的一个显示深发展日线的小程序,运行时要将

000001.day拷到当前目录.

```
#include <stdio.h>

#include <stdlib.h>

#include <conio.h>

typedef struct

{

unsigned long date;

unsigned long open;

unsigned long low;

unsigned long high;

unsigned long close;

unsigned long travl;

unsigned long traca;

char unuse[12];

} RECORD;

RECORD reco;

int readrec(FILE *);

void main()

{

FILE *fp;

if((fp = fopen("000001.day","rb")) == NULL)

{

printf("Error: Can't open 000001.DAY !\n");

exit(0);

}

readrec(fp);

fclose(fp);

if(getch()==0) getch();

exit(0);
```

```
}

int readrec(FILE *fp)

{

float fn;

while (! feof(fp))

{

fread(&reco,sizeof(RECORD),1,fp);

printf("%10lu ",~reco.date);

fn=float(~reco.open)/1000;

printf("%8.2f ",fn);

fn=float(~reco.low)/1000;

printf("%8.2f ",fn);

fn=float(~reco.high)/1000;

printf("%8.2f ",fn);

fn=float(~reco.close)/1000;

printf("%8.2f ",fn);

printf("%8lu ",~reco.travl);

printf("%8lu\n",~reco.traca);

if(kbhit()) break;

}

printf("\n");

return 0;

}
```

通达信数据格式

通达信日线

每条数据文件占32个字节。每四个字节代表一个变量。如下所示：

```
struct TDX_DAY
{
    int date;
    int open;    //元
    int high;
    int low;
    int close;
    float amount; //元
    int vol; //股 1手=100股
    int reservation;
} ;
```

股票数据文件名为: sh600036.day
每32个字节代表一天的股票数据
用股票数据文件的大小除以32个字节可以得出总共有多少天的数据

我以招商银行为例说明其数据格式:

日线数据存放在 C:\Program Files\国泰君安证券\超强版\vipdoc\sh\lday

文件名: sh600036.day

以下是数据文件图片显示.首页内显示不了图片,请点击标题进入看.

其每32个字节记录的是一日线数据:

其数据结构定义如下:

```
typedef struct mystructtag
{
   int date;
   int open;
   int high;
   int low;
   int close;
   float amount;
   int vol;
   int reservation;
} StockData;
```

这个数据是以上图片翻译的两行数据

日期 开牌价 最高价 最低价 收盘价 成交量

20020409 10.51 10.88 10.51 10.66 414108800
20020410 10.66 10.70 10.39 10.60 67945400

通达信五分钟线

```
struct TDX_5MIN
{
    unsigned short nianyue;
    unsigned short xiaoshifenzhong;
    float open;
    float high;
    float low;
    float close;
    float chengjiaoe;
    int chengjiaoliang;
    int reservation;
};
```

通达信的5分钟数据格式 *.lc5

示例数据：
32字节为单位：
CD 00 3F 02 33 33 0F 42-7B 14 11 42 66 66 0E 42
3D 0A 11 42 B0 2F F6 4B-A4 B2 0D 00 00 00 00 00

>>> struct.unpack('hhfffffii', buf)
(205, 575, 35.799999237060547, 36.270000457763672, 35.599998474121094, 36.259998321533203, 32268128.0, 897700, 0)

205 / 100 : 月
205 %100 : 日
575 / 60 : 小时
575 % 60 : 分钟
（实数）OPEN
（实数）LOW
（实数）HIGH
（实数）CLOSE
（实数）成交额
成交量
保留

通达信分时数据格式

通达信的zst的数据记录是每6508个字节为一天的数据,每26个字节为一个分钟的记录,这26个字节是这样分配的,时间占两个字节化为十进制为570的话表示9:30分(570/60=9.5)　下一个是占四个字节的叫现价,再下四个字节叫均价,另外还有两个字节为该分钟成交量(现在有可能已经改为四个字节),剩下的14个字节是预留的,那么大家会发现用我以前所有介绍的方法求出的现价和均价都是个天文数字,和股票价格相差甚远但又有规律似的,一般好像玛雅人才用到似的,我介绍一种最简单的办法给大家去求价格的对应关系,大家任找门股票把他的在整数位价格对应的四个字节的16进制和价格位记下来,如4元对应40800000　　其实这时你如果对zstli任意个16进制的价格x 想求出他的对应的真实价格,只要用过比例式就能求出,不过先别高兴去操作,因为并不是所有价格档次都是同个比例的,大家看下面的表对照一下,我已经帮大家总结到32的价格,可算到64块钱内个股票64和128元的。

该算法适用与所有通达信行情软件,如南方证券,鑫网通达信行情软件,并且价格的换算算法对通达信的day日数据里面的价格也适用

RMB	zst数据	zst数据10进制	1￥对应的16进制间隔	人民币间隔	1￥对应的10进制间隔
A	B	C	D	E	F
0	3F800000	1065353216	400000	1￥	4194304
4	40800000	1082130432	200000	1￥	2097152
8	41000000	1090519040	100000	1￥	1048576
16	41800000	1098907648	80000	1￥	524288
32	42240000	1109655552	40000	1￥	262144
		1118830592			

按比例算法求。设取到的数为X(化为十进制的了),那么(X-C)/real(x)=F/(1￥*1000)　　扩大一千倍,其实以上的C和F就是你可以通过判断逻辑求出取哪个对应的哪个C只有real(x)未知
移动方程两边可求出准确的价格(均价可达到0.001精度),求出后的十进制价格是放大了1000倍的

通达信目录结构

vipdoc:下载或缓存的历史数据目录
diary:投资日志目录
RemoteSH:缓存的上海F10
RemoteSZ:缓存的深圳F10
Ycinf缓存的公告消息
安装目录下的SZ.*,SH.*是缓存的盘中数据文件

T0002:个人信息目录,内有公式和自选股,个人设置等信息
Advhq.dat 星空图相关个性化数据
Block.cfg 板块设置文件
cbset.dat 筹码分析个性化数据
colwarn3.dat 行情栏目和预警个性化数据
colwarnTj.dat 条件预警个性化数据
CoolInfo.Txt 系统备忘录
Line.dat 画线工具数据
MyFavZX.dat 资讯收藏夹数据
newmodem.ini 交易客户端个性化数据
padinfo.dat 定制版面个性化数据
PriCS.dat,PriGS.dat,PriText.dat 公式相关数据
recentsearch.dat 最近资讯搜索数据
Scheme.dat 配色方案
tmptdx.css 临时网页CSS文件
user.ini 全局个性化数据
userfx.dat K线图个性化数据

[blocknew] 板块目录
[cache] 系统数据高速缓存
[zst_cache] 分时图数据高速缓存
[coolinfo] 系统备忘录目录
[Invest] 个人理财数据目录

通达信软件其他文件格式以及相关资料

分笔成交中的"B","S"标记
"B" 表示是主动性买单(Buy)
"S" 表示是主动性卖单(Sell)
无BS标记的表示是不明单,系统根据当时的叫买叫卖价无法得知是主动性买单还是卖单

分笔成交明细中的最右边的灰色数字表示的是什么？
交易所发布的行情中,每一个分笔并不是只有一笔成交,可能是几笔合成,深交所发布的数据有笔数信息,灰色数字就是该分笔数据中实
际上包含多少笔成交

分笔成交明细和行情信息中有的成交量为紫色，是什么意思？
表示是大单，缺省500手以上为大单，这个值可以通过“系统设置”->“参数1”->现量高亮成交量 来调整

F2分价表中的竞买率是什么含义
竞买率表示在此价位上成交的量中，主动性买量占的比率。

通达信分时图成交量柱状线颜色的含义?
当在系统设置中打开"分时图中成交量区间颜色显示"时,分时图中的成交量不再是单一的成交量颜色,而是有三种颜色:
红色表示成交量是价格上涨过程中成交的；绿色表示成交量是价格下跌过程中成交的；白色表示是价格不变过程中成交的量

右上角行情信息区的证券名称前有P,L标识是什么意思？
P表示此股本有机构评级数据。
L表示此股存在关联品种，比如有B股，可转债，H股或权证等等，点击之可以切换到相关的品种。

关于量比
量比是一个衡量相对成交量的指标，它是开市后每分钟的平均成交量与过去5 个交易日每分钟平均成交量之比。
量比数值大于1，说明当日每分钟的平均成交量大于过去5个交易日的平均数值，成交放大；
量比数值小于1，表明现在的成交比不上过去5日的平均水平，成交萎缩。

在分时图中，按/*键出现的量比图的含义:
若是突然出现放量，量比指标图会有一个向上突破，越陡说明放量越大；若出现缩量，量比指标会向下走。
大盘分时图上的黄线是什么线

大盘分时图上的白线是交易所发布的指数价线,黄线是软件商自行统计的不加权均线,也就是所有的成份股票按相同的权重进行统计计
算,一般来说,如果两线背离严重,说明大盘股和小盘股的涨跌差距较大
通达信多空红绿军的解释:

在状态栏上有两个方格条,左边为沪市的多空条,右边为深市的多空条.
多空条分两部分:
向左是涨的股票比例(为红色,如果为深红,表示涨停部分),
向右是跌的股票比例(为绿色,如果为深绿,表示跌停部分)

方格条下面有6种不同的符号在滚动:
红色向上的箭头:表示整个市场涨势在增加
红色向下的箭头:表示整个市场涨势在减弱
红色等于号:表示整个市场涨势保持持平
绿色向上的箭头:表示整个市场跌势在增加
绿色向下的箭头:表示整个市场跌势在减弱
绿色等于号:表示整个市场跌势保持持平

判断涨势和跌势有一个量化指标数据(即为领先值):
此数据大致上是按照当前市场上所有股票最新成交相对于上次成交是涨或是跌,涨多少或是跌多少等统计出来的.
如果此数据大于0,显示为红色.此数据大于上次统计的数据,显示为红色向上的箭头,反之,则显示为红色向下的箭头,相等则为等于号.
如果此数据小于0,显示为绿色.此数据小于上次统计的数据,显示为绿色向上的箭头,反之,则显示为绿色向下的箭头,相等则为等于号.
一些字段释义

量变幅度:
(期末成交量-期初成交量)/期初成交量

市场比:
表示当前区间内的该股票的总成交金额占所在市场(上海或深圳)的总成交金额的比例

权涨幅:
在热门板块中,按流通盘加权的涨幅平均值

财务计算公式

市盈率:
现价/((净利益*12/季报月份)/总股本)

净益率:
净利益/净资产

每股未分配:
未分配利润/总股本

每股收益:
净利益/总股本

每股净资产:
净资产/总股本

股东权益比:
净资产/总资产

每股公积金:
资本公积金/总股本

每股经营现金流:
经营活动现金流量/总股本

主营业务利润率:
主营利润/主营成本

速动比率:
(流动资产-存货)/流动负债

资产负债率:
总负债/总资产

几个内置指标的含义

AH: 高价突破点
NH: 卖点
CDP: 昨日中价
NL: 买点
AL: 低价突破点

多空平衡：多空的平衡价位点
如果适合作多,则有多头获利;多头止损
如果适合作空,则有空头回补;空头止损
活跃度表示某只股票的成交情况，基本上等于当天的成交笔数
强弱度表示其涨幅与大盘的涨幅之差

通达信复权模型:

通达信复权模型是建立在"股东财富不变"的原则上的。
交易所依据"股东财富不变"原则制定除权除息报价计算公式,目前沪深交易所除权除息报价的基本公式如下(在具体操作中可能会
有所变动):
除权(息)报价＝[(前收盘价-现金红利)＋配(新)股价格×流通股份变动比例]÷(1＋流通股份变动比例)

通达信复权分为向前复权和向后复权:
向前复权,就是保持现有价位不变,将以前的价格缩减,将除权前的K线向下平移,使图形吻合,保持股价走势的连续性。
向后复权,就是保持先前的价格不变,而将以后的价格增加。上面的例子采用的就是向后复权。
两者最明显的区别在于向前复权的当前周期报价和K线显示价格完全一致,而向后复权的报价大多低于K线显示价格。例如,某只股
票当前价格10元,在这之前曾经每10股送10股,前者除权后的价格仍是10元,后者则为20元。

复权是根据上市公司的权益分派、公积金转增股本、配股等情况和交易所的除权报价方案精确计算复权价格。其计算公式:

前复权:复权后价格＝[(复权前价格-现金红利)＋配(新)股价格×流通股份变动比例]÷(1＋流通股份变动比例)
后复权:复权后价格＝复权前价格×(1＋流通股份变动比例)-配(新)股价格×流通股份变动比例＋现金红利

通达信网上交易客户端的复权K线范围是所有从服务器端取得的数据,如果将分析股票的所有数据(从上市第一天开始)取到了本地,则
复权是基于所有数据的(数据的多少对后复权的当前价格有很大影响)。
通达信复权算法:

一、除权除息的概念
上市证券发生权益分派、公积金转增股本、配股等情况,交易所会在股权(债权)登记日(B股为最后交易日)次一交易日对该证券作
除权除息处理。

除权除息的基本思想就是"股东财富不变"原则,意即分红事项不应影响股东财富总额,这是符合基本财务原理的。依据此原则,交
易所在除权前后提供具有权威性的参照价格,作为证券交易的价格基准即除权除息报价。
在除权除息日交易所公布的前收盘是除权除息报价而非上一交易日收盘价,当日的涨跌幅以除权除息报价为基准计算,所以能够真
实反映股民相对于上一交易日的盈亏状况。
交易所依据"股东财富不变"原则制定除权除息报价计算公式,目前沪深交易所除权除息报价的基本公式如下(在具体操作中可能会
有所变动):
除权(息)报价＝[(前收盘价-现金红利)＋配(新)股价格×流通股份变动比例]÷(1＋流通股份变动比例)

二、复权的概念
除权、除息之后,股价随之产生了变化,往往在股价走势图上出现向下的跳空缺口,但股东的实际资产并没有变化。如:10元的股票
,10送10之后除权报价为5元,但实际还是相当于10元。这种情况可能会影响部分投资者的正确判断,看似这个价位很低,但有可能是一
个历史高位,在股票分析软件中还会影响到技术指标的准确性。
所谓复权就是对股价和成交量进行权息修复,按照股票的实际涨跌绘制股价走势图,并把成交量调整为相同的股本口径。例如某股
票除权前日流通盘为5000万股,价格为10元,成交量为500万股,换手率为10%,10送10之后除权报价为5元,流通盘为1亿股,除权当日走出
填权行情,收盘于5.5元,上涨10%,成交量为1000万股,换手率也是10%(和前一交易日相比具有同样的成交量水平)。复权处理后股价为
11元,相对于前一日的10元上涨了10%,成交量为500万股,这样在股价走势图上真实反映了股价涨跌,同时成交量在除权前后也具有可比
性。

三、向前复权和向后复权
向前复权,就是保持现有价位不变,将以前的价格缩减,将除权前的K线向下平移,使图形吻合,保持股价走势的连续性。
向后复权,就是保持先前的价格不变,而将以后的价格增加。上面的例子采用的就是向后复权。
两者最明显的区别在于向前复权的当前周期报价和K线显示价格完全一致,而向后复权的报价大多低于K线显示价格。例如,某只股
票当前价格10元,在这之前曾经每10股送10股,前者除权后的价格仍是10元,后者则为20元。

四、自动复权和精确复权
所谓自动除权,指股票软件自动确定当日是否有除权发生,根据今日收到的昨收盘和上一交易日的收盘价对比,若二者不等,则能肯
定今天有除权,进而推算送配方案,进行复权处理。这种方法有很多问题,不能做到准确复权。
精确复权是根据上市公司的权益分派、公积金转增股本、配股等情况和交易所的除权报价方案精确计算复权价格。精确复权的计
算公式:

前复权:复权后价格＝[(复权前价格-现金红利)＋配(新)股价格×流通股份变动比例]÷(1＋流通股份变动比例)
后复权:复权后价格＝复权前价格×(1＋流通股份变动比例)-配(新)股价格×流通股份变动比例＋现金红利
T0002目录下的文件说明

如果重装到一个新位置，只需将整个T0002拷贝过去就可以了,所有的个性化数据都在此目录下

Advhq.dat 星空图相关个性化数据
Block.cfg 板块设置文件
cbset.dat 筹码分析个性化数据

CoolInfo.Txt 系统备忘录
Line.dat 画线工具数据
MyFavZX.dat 资讯收藏夹数据
newmodem.ini 交易客户端个性化数据
padinfo.dat 定制版面个性化数据
PriCS.dat,PriGS.dat,PriText.dat 公式相关数据
recentsearch.dat 最近资讯搜索数据
Scheme.dat 配色方案
tmptdx.css 临时网页CSS文件
user.ini 全局个性化数据
userfx.dat K线图个性化数据
mark.dat 标识信息的存盘文件
以下文件与设置的预警信息有关：
Col_warn.dat
Col_warn_self.dat
Col_warn2.dat
ColwarnTj.dat

[blocknew] 板块目录
[cache] 系统数据高速缓存
[zst_cache] 分时图数据高速缓存
[coolinfo] 系统备忘录目录
[Invest] 个人理财数据目录
[PAD] 定制牘面存盘文件

通达信自选股的格式

自选股存放在安装目录T0002\blocknew\ZXG.blk文件
此文件是一个文本文件,一行代表一只股票,
每行的第一个字符表示市场,'0'表示深圳,'1'表示上海,跟后的字符串表示股票代码.
比如1999999表示上证指数,0000002表示深万科

通达信新版本增加的权证字段：

杠杆比率：
标的证券价格/（权证价格÷行权比例）

内在价值：
权证价格由内在价值和时间价值两部分组成。当标的证券价格高于行权价时，内在价值为两者之差；而当标的证券价格低于行权价
时，内在价值为零。但如果权证尚没有到期，标的证券价格还有机会高于行权价，因此权证仍具有市场价值，这种价值就是时间价值
。
认股权证内在价值＝(标的证券价格－行权价)*行权比例，若标的证券价格<=行权价，则内在价值为0；认沽权证内在价值＝(行权价
－标的证券价格)*行权比例，若标的证券价格>=行权价，则内在价值为 0。

时间价值：
时间价值＝权证实际市场价格－内在价值，任何权证的价格都是内在价值和时间价值的和，投资者可以时时计算内在价值，但时间
价值随着权证上市不断减少，所以投资者去判断权证价格时，应该注意随着行权时间的临近，权证价格必然要不断接近内在价值。

溢价率：
溢价率就是在权证到期前，正股价格需要变动多少百分比才可让权证投资者在到期日实现打和。溢价率是量度权证风险高低的其中
一个数据，溢价率愈高, 打和愈不容易。
认购权证溢价率＝[(行权价+认购权证价格/行权比例) / 标的证券价格 — 1]×100%
认沽权证溢价率＝[1 — (行权价－认沽权证价格/行权比例) / 标的证券价格]×100%

隐含波动率：(尚未加入)
香港市场称为“引伸波幅”，引伸波幅是市场对相关资产在未来一段时间内的波动性的预期，当引伸波幅上升时，认股权证的价格
会调高，而当引伸波幅下跌时，认股权证的价格将调低。投资者应该在引伸波幅较低的时候买入，引伸波幅较高的时候卖出。

行权价格和行权比例的变动：
新行权价格=原行权价格×（标的证券除权日参考价/除权前一日标的证券收盘价)；
新行权比例=原行权比例×（除权前一日标的证券收盘价/标的证券除权价)。

权证涨跌幅：
权证涨幅价格＝权证前一日收盘价格＋(标的证券当日涨幅价格-标的证券前一日收盘价)×125%×行权比例；
权证跌幅价格＝权证前一日收盘价格-(标的证券前一日收盘价-标的证券当日跌幅价格)×125%×行权比例，当计算结果小于等于
零时，权证跌幅价格为零。

缺省MA均线只有四条,想修改成六条怎么做?

直接敲MA2,或者在主图中点右键,选择"主图坐标",再在里面选择MA2

上证换手:

(上海市场的A,B股和封闭式基金的成交量)/(上海市场的A,B股和封闭式基金的流通股本)

深证换手亦然

通达信中几个特别字符的意义:

L:表示存在关联品种,比如有权证,B股,转债或H股等

P:表示有评级信息

T:表示用户对此股票进行了文字标记

怎样进行指标排序功能?

使用"历史行情报表"功能可以实现任一天的指标排序
前提是要下载完整的日线数据

在报价菜单中选择历史行情报表或用键盘精灵按.401进入

飞狐数据格式

飞狐目录结构

DATA\ 分市场存放行情历史数据及个股F10资料，其中：
$$\ 存放板块指数、投资指数历史数据
\Day*.day 同名个股日线数据文件
\F10*.txt 同名个股F10资料文本文件
StkData.sif 证券信息文件，含证券代码表、财务数据、除权数据等
StkBlock\
StkBlock.ini 分类板块配置文件，可以进行手工调整，如改变板块的排列顺序等；
*.sbk 同名板块文件
*.cbk 板块指数定义文件
System\ 存放各种系统数据文件
SysFml.fal 系统公式文件
SysFml.oal 原始系统公式文件，用于恢复被修改过的系统公式
Fee.dat 分类证券交易费率文件
Lunar.dat 用于快速计算阴历时间
Option.ini 系统选项文件，现暂只有移动成本计算的设置信息
HqOnline.ini 互联网行情服务器配置文件
F10Fit.txt 从F10资料提取财务数据的摸板文件
User\ 存放各种用户数据文件，用于保存指标公式、界面布局、组合条件、预警、画线、解盘系统等。
UserFml.fal 所有自编公式文件
Alert.alt 预警系统文件
Draw.drw 画线文件
Default.fjp 缺省解盘文件
*.fjp 解盘文件
*.alg 导出的公式文件
*.fcc 组合条件文件
*.lyt 窗口布局文件
*.bkc 自动板块设置文件
*.dlg 下载选项保存文件
*.fct 计算器算式文件
Params\ 保存公式参数信息
FmlParams.dfp 保存用户调整过的公式参数
FmlDLL\
公式扩展调用的DLL文件须放此目录
Invest\ 存放投资帐户文件
*.ivs 同名投资帐户的数据文件
*.imm 同名投资帐户的备忘录文件
InfoSys\
Info.ini飞狐资讯浏览器（FoxIE, Fox Infomation Explorer）的配置文件，
可手工增改此文件，使URL指向自己感兴趣的地方
Download\ 存放批量下载回来的文件，如2000.qda、StkData.sif等，
*.qda 日线数据补数文件
F10资料直接下载解压到系统F10资料目录了

TradeTest\ 存放交易测试文件（尚未用到）
*.ttm 交易测试模型文件
*.ttr 交易测试结果文件

飞狐日线数据格式

飞狐的日线文件的下载地址是
http://down.88158.cn/

下载了的日线文件的后缀名一般是打包的rar，解开之后得到Quote.QDA

下面是这个Quote.QDA的数据格式：
文件头：
int32 header = 0xFFFFFFE2
int32 type　= 0×00000101
int32 stockcount

用UE看就是E2FFFFFF 01010000[stockcount]，这里的header是区分一个文件是日线数据还是其他数据的标志，如果是五分钟线，header=0xFFFFFFE1。

stockcount就是包含的股票数量了。

文件头12个字节之后就是数据区了。

数据区按股票排列，一共有stockcount个股票，每个股票的格式如下：

股票日线：

char[12] stockid 股票ID

char[12] stockname 股票中文名称

int32 daycount 该股票一共有日线的个数

每日数据[daycount] 一个每日数据的数组

每日数据的格式：

int32 UTC_time UTC 表示的东八区时间
float open 开盘价
float high 最高价
float low 最低价
float close 收盘价
float volume 成交量
float amount 成交金额
int32 dealcount 成交次数

简单点说，就是先按股票读，在每个股票描述之后是该股票的日线数据。该股票所有日线数据结束之后是下一个股票的描述信息。

其他股票软件数据格式

恒生日线文件数据结构

沪市日线文件路径：默认在HSNEW\DATA\SHASE\DAY下。
深市日线文件路径：默认在HSNEW\DATA\SZNSE\DAY下。

日线文件命名规则：股票代码.DAY

每个日K线为40字节，具体如下：

```
Date:LongInt; //日期
OPen:LongInt; //开盘(元/1000)
High:LongInt; //最高价(元/1000)
Low:LongInt; //最低价(元/1000)
Close:LongInt; //收盘(元/1000)
Money:LongInt; //成交额(千元)
Volume:LongInt; //成交量(手)
Nouse1:LongInt; //没用
Nouse2:LongInt; //没用
Nouse3:LongInt; //没用
```

海融动态“闪电版”日线数据格式

上海日线存储路径为:\hrdyn\shday,文件扩展名为:.day
深圳日线存储路径为:\hrdyn\szday,文件扩展名为:.day

以深发展日线(0001.day)为例:

```
0000h: F8 30 31 01 CD CC 88 41-AE 47 89 41 00 00 86 41
0010h: 71 3D 86 41 24 02 20 00-62 26 87 41 00 00 00 00
0020h: 00 00 00 00 00 00 00 00-F9 30 31 01 66 66 86 41
0030h: CB A1 88 41 14 AE 85 41-A4 70 87 41 9C C5 11 00
0040h: AA C3 86 41 00 00 00 00-00 00 00 00 00 00 00 00
```

```
起止地址 数据内容 数据含义 数据类型
0000 - 0003 F8 30 31 01 日 期 Integer
0004 - 0007 CD CC 88 41 开盘价 Single
0008 - 000B AE 47 89 41 最高价 Single
```

000C - 000F 00 00 86 41 最低价 Single
0010 - 0013 71 3D 86 41 收盘价 Single
0014 - 0017 24 02 20 00 成交量 Integer
0018 - 001B 62 26 87 41 均 价 Single
001C - 001F 00 00 00 00 前收盘 Single
0020 - 0021 00 00 竞 Word
0022 - 0023 00 00 量
Word
0024 - 0027 00 00 00 00 成交笔数 Integer

注：
1）起止地址、数据内容为十六进制，数据类型为 Delphi 下之定义。
2）从0028h开始每40byte为一条股票数据记录，含义如上表0000h - 0027h所示；

汇金数据格式

上海日线存储路径为:\hjin\exe\lineday\sh,文件扩展名为:.psd
上海周线存储路径为:\hjin\exe\lineweek\sh,文件扩展名为: .psw
上海月线存储路径为:\hjin\exe\linemon\sh,文件扩展名为: .psm
深圳日线存储路径为:\hjin\exe\lineday\sz
深圳周线存储路径为:\hjin\exe\lineweek\sz
深圳月线存储路径为:\hjin\exe\linemon\sz
以深发展日线为例:
1A76:0100 D6 CD 2F 01 EC D1 86 42-EC D1 86 42 EC D1 86 42
1A76:0110 EC D1 86 42 7B F9 86 42-14 E6 00 00 20 D7 CD 2F
1A76:0120 01 CD CC 84 42 CD CC 84-42 CD CC 84 42 CD CC 84
1A76:0130 42 D5 04 85 42 D0 52 00-00 20 D8 CD 2F 01 54 23
1A76:0140 84 42 54 23 84 42 54 23-84 42 54 23 84 42 9E E7
1A76:0150 83 42 A0 41 00 00 20
每一条记录的长度为29字节:
1-4字节为日期,D6 CD 2F 01转换为十进制是:19910102
5-8字节为开盘价
9-12字节为最高价
13-16字节为最低价
17-20字节为收盘价
21-24字节为均价
25-28字节为成交量(股)
其余1字节未使用
另:周线,月线格式与日线格式一致.
下面是我用C语言编的一个显示深发展日线的小程序,运行时要将
sz0001.psd拷到当前目录.

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
typedef struct {
unsigned long date; //日期
float open; //开盘价
float high; //最高价
float low; //最低价
float close; //收盘价
```

```
float cavl; //均价
unsigned long travl; //成交量
char unuse;
} RECORD;
RECORD reco;
int readrec(FILE *);
void main()
{
FILE *fp;
if((fp = fopen("sz0001.psd","rb")) == NULL) // 打开深发展日线
{ printf("Error: Can^t open SZ0001.PSD !\n");
exit(0); }
readrec(fp);
fclose(fp);
if(getch()==0) getch();
exit(0);
}
int readrec(FILE *fp)
{
while (! feof(fp)) {
fread(&reco,sizeof(RECORD),1,fp);
printf("%10lu ",reco.date);
printf("%8.2f ",reco.open);
printf("%8.2f ",reco.high);
printf("%8.2f ",reco.low);
printf("%8.2f ",reco.close);
printf("%8.2f ",reco.cavl);
printf("%8lu\n",reco.travl);
}
printf("\n");
return 0;
}
```

慧眼日线文件数据结构

慧眼日线文件数据结构

日线文件路径：默认在\Stkdt下。

日线文件命名规则：股票代码.DAT

每个日K线为28字节，具体如下：

```
Date:LongInt; //日期
OPen:LongInt; //开盘(元/100)
High:LongInt; //最高价(元/100)
Low:LongInt; //最低价(元/100)
Close:LongInt; //收盘(元/100)
App:LongInt; //涨跌(收盘价之差)
Volume:LongInt; //成交量(手)
```

慧眼代码表文件数据结构

慧眼98静态版的lstf.dat的数据结构(本文件是列表与日线无关)

以4位整型为单位。

第四个为日期。前100个字节为头信息。好像不能动。每个股票为60字节

为代码(ha600001) 为名称
开(分) 高(分) 低(分) 收(分)
涨跌(分) 成交量(手) ？ 00 00 10 16(0) ？ 0
？ 0 ？ 0 ？ 0 ----------

一个长整型的各位的含义

即与00 00 11 13异或的结果

显示值 (1-H)2^28 (1-L)2^24 (2-H)2^20 (2-L)2^16 (3-H)2^12 (3-L)2^8 (4-H)2^4 (4-L)2^0
0 268435456 16777216 1048567 65536 1*4096 1*256 1*16 3
1 1 1 1 1 0 0 0 2
2 2 2 2 2 3 3 3 1
3 3 3 3 3 2 2 2 0
4 4 4 4 4 5 5 5 7
5 5 5 5 5 4 4 4 6
6 6 6 6 6 7 7 7 5
7 7 7 7 7 6 6 6 4
8 8 8 8 8 9 9 9 11
9 9 9 9 9 8 8 8 10
a a a a a 11 11 11 9
b b b b b 10 10 10 8
c c c c c 13 13 13 15
d d d d d 12 12 12 14
e e e e e 15 15 15 13
f f f f f 14 14 14 12

如果涨跌幅是负的话，则用FF FF FF FF 减去用上面算法算出的数再减1

显示的成交量可以与内建的流通量一起由慧眼算换手率。

同花顺日线数据格式

日数据格式为
64个字节的文件头,另外每48个字节为一天纪录
日期 32位 4个字节
开盘 32位无符号整型 4个字节
最高 32位无符号整型 4个字节
最低 32位无符号整型 4个字节
收盘 32位无符号整型 4个字节
跳过 32位 4个字节
成交量 32位 4个字节
跳过 20个字节

以上跳过部分只有最后4个字节未使用
跳过部分包含数值,涨幅,振幅,总成交金额,换手率。

如何编程实现解析同花顺日线数据格式

日数据为48个字节一天,文件头为64个字节n = (fs.Length - 64) / 48 '可得天数,详细代码如下
VB.net 2005

```
Case 3 '同花顺
n = (fs.Length - 64) / 48 '文件头占64个字节
LoadDayData = n
ReDim dr(n)
For i = 1 To 8
br.ReadInt64() '读8次64位内存块以越过文件头
Next
For i = 1 To n
Dim d As Integer = br.ReadInt32()
j = j + 1
dr(j).stockdate = DateSerial(d / 10000, (d Mod 10000) / 100, d Mod 100) '日期占4个字节
dr(j).openor = (br.ReadUInt32 And &HFFFFFFFF) / 1000 '用无符号32位整型,目前只发现B0开头,所以只需除以1000,不判断,但最好做个异常分支
dr(j).hightor = (br.ReadUInt32 And &HFFFFFFFF) / 1000
dr(j).lowor = (br.ReadUInt32 And &HFFFFFFFF) / 1000
dr(j).endor = (br.ReadUInt32 And &HFFFFFFFF) / 1000
br.ReadInt32() '越过数据,不影响对数据基本数据的获取
dr(j).changor = br.ReadInt32 '(单位手数),成交量,也是32位,可以用无符号整型去读
If dr(j).changor < 0 Then '以B0 开头
dr(j).changor = (dr(j).changor And &HFFFFFFFF) / 1000
ElseIf dr(j).changor > 0 Then '这里需要改进 目前发现 90开头
dr(j).changor = dr(j).changor / 100
Else
' ......
End If
br.ReadInt32() '把纪录尾端读完,共20个字节
br.ReadInt64()
br.ReadInt64()
'其实以上越过的数据包含信息为数值,涨幅,振幅,总金额,换手,其实一个日纪录48个字节就只要4个字节未用到的了
Next
case 4 '大智慧level2
```

天网数据格式

天网数据数据结构

天网日线文件数据结构

沪市日线文件路径：默认在YYY\DATA\SHASE\DAY下。
深市日线文件路径：默认在YYY\DATA\SZNSE\DAY下。

日线文件命名规则：股票代码.DAY

每个日K线为40字节，具体如下：

Date:LongInt; //日期
OPen:LongInt; //开盘(元/1000)
High:LongInt; //最高价(元/1000)
Low:LongInt; //最低价(元/1000)
Close:LongInt; //收盘(元/1000)
Money:LongInt; //没用
Volume:LongInt; //成交量(手)
Nouse1:LongInt; //没用
Nouse2:LongInt; //没用
Nouse3:LongInt; //没用

图文卡与钱龙分析软件接口规范

第一章 系统概述

系统的物理结构

股票分析软件 Lonsd.exe(单机版)

驱动程序 Driver.exe(TSR)

操作系统 DOS3.30 or High

硬件平台 PC + 数据接收机

数据接收机可有多种实现方式，典型的有：

VBI（图文电视）方式

RDS（调频副载波）方式

MODEM（调制解调器）方式

可视图文方式

X.25方式

股票分析软件Lonsd.exe通过驱动程序Diver.exe访问股票的行情数据，与不同的接收方式或硬件平台无关，而每一种数据接收机均需一个特殊的驱动程序Diver.exe以保证数据的正常接收。驱动程序Diver.exe为一常驻内存程序（TSR），负责实时数据的接收。所有的数据接收机的硬件生产厂商需提供用户针对该接收机的驱动程序Diver.exe，并保证能接收到本规范所定义的钱龙动态股票分析软件所需要的数据项，用户即可使用钱龙公司出品的最新股票分析单机版软件。实时数据发射端和数据接收机之间的通讯协定由数据接收机的硬件生产厂商决定，只须满足速度及稳定性要求即可。

系统的配置需求

CPU 80386、80486 or Pretium

Memory >=2M, Free Memory>=1.6M(建议>=4M)

Display Standard VGA 640*480 16色

DOS VER3.30 or High(建议使用SmartDrv.exe)

TLOOS 钱龙公司天生赢家Winnegro专用软件狗一个

接收速度 建议>=1 记录数/秒

第二章 动态单机版工作流程

驱动程序Diver.exe和分析软件Lonsd.exe的共用数据区

共用数据区由BIOS通讯区和行情数据区两部分组成。BIOS通讯区由硬件类型、硬件机号、行情类型、数据区地址（段地址和偏移量），数据区记录首指针、尾指针和Diver驻留标志等元素组成。行情数据区以队列方式存放，每条启示为96个字节，共有256条记录，记录格式见第三章。

驱动程序Diver.exe的工作流程

非驻留部分：

初始化，检查硬件-数据接收机是否存在。

接收所有股票的基本资料，建立证券名称对照表（需存盘，含各种指数），证券名称对照表见第三章，证券名称对照表按类别（type）排序，相同类别则按股票代号排序。

初始化BIOS通讯区，置硬件类型、硬件机号和行情类型、数据区段地址、偏移置和Diver驻留标志均需为0。

驻留部分：

Driver驻留标志置为0xaa，检查分析软件是否启动，即判断数据区内存地址是否已置好。

若好，则接收数据。若当前数据首指针为n，则将行情数据写至（数据区内存地址+n*88)位置，然后置数据首指针置为n+1。（当n=255时，数据首指针置为0）。

若数据区内存地址为0，则停止接收数据。

分析软件Lonsd.exe的工作流程

检查是否已安装驱动程序，若是则申请数据区内存，置数据区内存地址。

将数据区首指针和尾指针置为0。

接收股票行情数据：判断首指针是否等于尾指针，若不等，设当前尾指针为m，则从（数据区内存地址+m*88)读一条记录，然后置数据尾指针置为m+1。（当m=255时，数据尾指针置为0）。

进入分析系统，循环处理第3条。

退出分析软件后将数据区内存地址重新置为0。

第三章 数据区、数据包及数据文件格式

BIOS通讯区数据表

硬件类型 0040:00f0 byte 1 Driver 不能为0

硬件机号 0040:00f1-00f5 5byte 5 Driver

行情类型 0040:00f6-00f7 16bit 1 Driver 最多为16种

数据区段地址 0040:00f8-00f9 word 1 Lonsd

数据区偏移地址 0040:00fa-00fb word 1 Lonsd

记录首指针 0040:00fc byte 1 Lonsd 由Driver改变

记录尾指针 0040:00fd byte 1 Lonsd 由Lonsd改变

汇金指标 0040:00fe byte 1 无

Driver驻留标志 0040:00ff byte 1 Driver 已驻留为0xaa

总计 0040:00f0-00ff 16

注：1、当汇金指标的值<=0f时，其值等于0，新股增加采用插入方式，其值等于1，新股增加采用代码表的方式，即使用exe目录下的NameTbl.Sha、NameTbl.Sza。

2、当汇金指标的值>0f时，则表示接收行情的方式当ax=0时，使用软中断接收行情，当ax=1时，使用硬中断接收行情0，并使用exe目录下的NameTbl.Sha、NameTbl.Sza。

指数数据包

数据包类型 type char 1 证券类型

证券代码 code(6) char 6

昨日收盘 close long 4

今日开盘 open long 4

今日最高 high long 4

今日最低 low long 4

今日最新 new long 4

总买盘量 vbuy long 4 可忽略

总卖盘量 bsell long 4 可忽略

总成交量 volume long 4 单位：百股

成交金额 amount long 4 单位：百元

总家数 total int 2 可忽略

日期 date long 4 可忽略

时间 time Long 4 可忽略

保留 reserved(34) Char 34

校验码 chksum Char 1

股票名称 name Char 8

总计 96

个股数据包

数据包类型 type Char 1 证券类型

证券代码 code(6) Char 6

昨日收盘 close Long 4

今日开盘 open Long 4

今日最高 high Long 4

今日最低 low Long 4

今日最新 new Long 4

买盘价1 pbuy1 Long 4

买盘量1 vbuy1 Long 4

买盘价2 pbuy2 Long 4

买盘量2 vbuy2 Long 4

买盘价3 pbuy3 Long 4

买盘量3 vbuy3 Long 4

卖盘价1 psell1 Long 4

卖盘量1 vsell1 Long 4

卖盘价2 psell2 Long 4

卖盘量2 vsell2 Long 4

卖盘价3 psell3 Long 4

卖盘量3 vsell3 Long 4

总成交量 volume Long 4 单位：股

成交金额 amount Long 4 单位：元

校验码 chksum Char 1

股票名称 name Char 8

总计　　96

信息数据包

数据包类型 type Char 1

时间 time Int 2 如12：59表示为1259

流水号 sieralno Int 2 信息编号（0-9999）

提供者 provider Char 14 信息来源

总块数 totblock Int 2

块号 blockno Int 2 块号=0时为信息标题

信息内容 content(64) Char 64 每块信息为64字节

校验码 chksum Char 1

总计　　88

证券名称对照表NameTbl.Sha NameTbl.Szn

证券类型 type Char 1 Bit7为标志位

证券代码 code(6) Char 6 不满8位填空格

证券名称 name(8) Char 8

最小交易单位 unit Char 1 每手股数/10

总计　　16

注意事项

所有股票的价格及指数的值均为实际值乘1000。股票的成交金额以元为单位，成交量（买卖盘）以股数为单位。指数的成交金额以百元为单位，成交量（买卖盘）以百股为单位。

日期的表示：如1996年1月1日为19960101（长整型）。

B股的价格单位：上交所为美元，深交所为港元，因此在计算B股指数的成交金额时必须乘上汇率，否则会导致B股成交金额和大盘成交金额有误差。

信息数据包块号=0时的内容为该信息的标题。信息的正文块号从1开始。

两个证券名称对照表文件的目录分别为\ML30\DATA\SHASE\NameTbl.Sha和\ML30\DATA\SZNSE\NameTbl.Sza。

Type作为证券类型时，Bit7为标志位，Bit7=1时忽略该股票。

Lonsd对每一个行情来源（上交所或深交所）最多能处理16个指数和560个股票。

目前Lonsd 读取行情记录时暂不对校验码(chksum)做检查。

附录一 实时行情种类

Byte 0: 0040:00f6

Bit7 Bit6 Bit5 Bit4 Bit3 Bit2 Bit1 Bit0

保留 保留 保留 保留 保留 保留 深证所 上证所

Reserved Reserved Reserved Reserved Reserved Reserved SZNSE SHASE

Byte 1: 0040:00f7

Bit7 Bit6 Bit5 Bit4 Bit3 Bit2 Bit1 Bit0

保留 保留 保留 保留 保留 保留 保留 保留

Reserved Reserved Reserved Reserved Reserved Reserved Reserved Reserved

附录二 数据包类型/证券类型

上海指数 0x10

上海A股 0x11

上海B股 0x12

上海债券 0x13

上海权证 0x14

上海红利 0x15

上海配股 0x16

上海证交所信息 0x1f

深圳指数 0x20

深圳A股 0x21

深圳B股 0x22

深圳债券 0x23

深圳权证 0x24

深圳红利 0x25

深圳配股 0x26

深圳证交所信息 0x2f

综合财经信息 0x05

空包 0xff

注：法人股系统及各期货交易上市的期货行情资料待日后（V3.10）增加。

附录三 证券代码（code)

个股的代码

个股的代码即为个股的股票代码，深圳证券交易所上市的股票的代码前4位有效，后2位填空格。

指数的代码

上证综合指数 999999

上证A股指数 999998

上证B股指数 999997

上证工业股指数 999996

上证商业股指数 999995

上证地产股指数 999994

上证公用事业股指数 999993

上证综合股指数 999992

上证30指数 999991

深圳成份股指数 9996

深圳成份A股指数 9995

深圳成份B股指数 9994

深证工业股指数 9993

深证商业股指数 9992

深证金融股指数 9991

深证地产股指数 9990

深证公用事业股指数 9989

深证综合股指数 9988

深证基金指数 9987

深证综合指数 9999

深证综合A股指数 9998

深证综合B股指数 9997

天亿日线文件数据结构

沪市日线文件路径：默认在\DATA\SHASE\DAY下。
深市日线文件路径：默认在\DATA\SZNSE\DAY下。

日线文件命名规则：股票代码.DAY

每个日K线为40字节，具体如下：

```
Date:LongInt; //日期
OPen:LongInt; //开盘(元/1000)
High:LongInt; //最高价(元/1000)
Low:LongInt; //最低价(元/1000)
Close:LongInt; //收盘(元/1000)
Money:LongInt; //成交额(千元)
Volume:LongInt; //成交量(手)
Nouse1:LongInt; //没用
Nouse2:LongInt; //没用
Nouse3:LongInt; //没用
```

投资家日线文件数据结构

投资家数据数据结构

每28个字节为一日。
Date:LongInt; //日期(距1970.01.01的秒数)
OPen:LongInt; //开盘(元)
Close:LongInt; //收盘(元)
High:LongInt; //最高价(元)
Low:LongInt; //最低价(元)
Volume:LongInt; //成交量(股)
Money:LongInt; //成交额(元)

日期，x/86400取余后如果余数大于等于57600则加1(取整方法为如果大于16:00收盘时间则加1，否则不加1，视为昨天。)即为1970-01-01到当前的天数。分析家则不用这样都是整数。 开盘价(算法同分析家) 收盘价(算法同分析家) 最高价(算法同分析家)
最低价(算法同分析家) 成交量(股)(算法同分析家) 成交额(元)(算法同分析家)从钱龙转过来正确，但自动接收好像不对为0

文华期货数据格式

每条记录读取4x9=36字节，各段字节如下
2A A6 39 40 日期 2004年2月23 存在86400基数
00 20 F0 44 开盘 F0 20 为61472 除32 1921
00 20 F1 44 收盘 F1 20 为61728 除32 1929
00 20 F2 44 最高 F2 20 为61984 除32 1937
00 60 F0 44 最低 F0 60 为61536 除32 1923
00 ED B9 47 仓单
80 7A 1E 48 量
6C 51 F1 44
EC B6 FE 3E

37 个字节一个记录，前4*7 = 28个字节如下:

```
struct WHDay
{
    time_t Date;          //日期
    float Open;           //开盘价
    float Close;          //收盘价
    float High;           //最高价
    float Low;            //最低价
    float Vol;            //成交量
    float Position;      //持仓量
} wenhuaday;
```

佛郎全自动股票交易系统3.0版本日线数据格式

此版本数据格式和大智慧日线数据格式相近，大家参考大智慧的日线数据结构即可出来。

基本有三个结构体构成

1. 全局结构体 （0x00 - 0x09h 共16个字节）

2. 索引结构体 （从0x10h开始，数目由第一个结构体提供）

3.日线结构体（数据从0x40010h开发）

剩下的请参照大智慧3日线结构。