# 速子信息极速行情系统

V1.1

TachyonINFO
FASTER IS BETTER

# 一、版权申明



# 二、修订记录

| 日期 | 撰稿人 | 备注 | 版本 |
| --- | --- | --- | --- |
| 20211115 | 速子信息 | 初版，SSE行情快照介绍 | V1.0 |
| 20211118 | 速子信息 | 添加SZSE行情快照、逐笔委托、逐笔成交 | V1.1 |

# 三、协议介绍

速子信息极速行情系统（以下简称TIMD）是基于交易所组播网络分发机制的快速行情系统，TIMD充分利用了交易所组播行情机制，采用一层交换机和快速解码的处理方式，实现了行情系统的加速。

TIMD无需库直接利用socket加入组播组，就可以接收转发的行情。转发的行情结构体，您可以参考本文档行情字段明细。接收行情的demo，可参考本文档DEMO。

# 四、行情连接地址

TIMD现支持5种组播行情的接收，相关的行情类目及表格请参考。

下表中IP地址及端口为TIMD推荐配置地址及端口，**实际地址端口请以连接服务端实际配置为准。**

| 行情类目 | IP地址 | 端口 |
| --- | --- | --- |
| 上海证券交易所（SSE） 行情快照-L1 | 239.20.30.90 | 10908 |
| 深圳证券交易所（SZSE）行情快照-L1 | 239.20.30.100 | 10910 |
| 深圳证券交易所（SZSE）行情快照-L2 | 239.20.30.101 | 10912 |
| 深圳证券交易所（SZSE）逐笔委托 | 239.20.30.102 | 10914 |
| 深圳证券交易所（SZSE）逐笔成交 | 239.20.30.103 | 10916 |

# 五、行情字段明细

## 5.1 快照行情-SSE-L1（312L）

### 5.1.1 快照字段对照表

下表为**上海证券交易所level1快照行情**与交易所原始字段的对照表格。

| 编号 | TIMD字段 | 交易所字段 | TIMD类型 | 交易所类型 | 长度 | 偏移量 | 字段详情 | 备注 |
|---|---|---|---|---|---|---|---|---|
| 1 | MdType | MDStreamID | char | char[5] | 1 | 0 | 1 = 指数类产品<br>2 = 股票（A、B 股）类产品<br>3 = 债券类产品<br>4 = 基金类产品（含公募REITs）<br>5 = 国债预发行产品<br>6 = 盘后固定价格<br>7 = 期权类产品<br><br>《行情类目详情参考表》请参考5.1.2字段说明第5条 | 行情类别 |
| 2 | UpdateTime | LastUpdateTime | uint32 | uint32 | 4 | 1 | 最新更新时间 HHMMSSsss | 最新更新时间 |
| 3 | SecurityID | SecurityID | char[10] | char[8] | 10 | 5 | 左对齐，多余字符为0 | 证券代码 |
| 4 | PreClosePrice | PreClosePx | double | uint64, N13(5) | 8 | 15 | | 昨收价 |
| 5 | NumTrades | NumTrades | longlong64 | uint64, N16 | 8 | 23 | | 成交笔数 |
| 6 | TotalVolumeTrade | TotalVolumeTraded | double | uint64, N16 | 8 | 31 | | 成交数量 |
| 7 | TotalValueTrade | TotalValueTraded | double | uint64, N16(2) | 8 | 39 | | 成交总金额 |
| 8 | LastPrice | MDEntryPx | double | uint64, N13(5) | 8 | 47 | 当MDEntryType = 2（指数行情时，MDEntryType = 3） | 最新价 |
| 9 | HighestPrice | MDEntryPx | double | uint64, N13(5) | 8 | 55 | 当MDEntryType = 4 | 开盘价 |
| 10 | LowestPrice | MDEntryPx | double | uint64, N13(5) | 8 | 63 | 当MDEntryType = 7 | 最高价 |
| 11 | ClosePrice | MDEntryPx | double | uint64, N13(5) | 8 | 71 | 当MDEntryType = 8 | 最低价 |
| 12 | SettlePrice | MDEntryPx | double | uint64, N13(5) | 8 | 79 | 当MDEntryType = 5 | 今收价 |
| 13 | PreSettlePrice | MDEntryPx | double | uint64, N13(5) | 8 | 87 | 当MDEntryType = 6 | 今结算价 |
| 14 | Iopv | MDEntryPx | double | uint64, N13(5) | 8 | 95 | 当MDEntryType = z1 | 昨结算价 |
| 15 | Prelopv | MDEntryPx | double | uint64, N13(5) | 8 | 103 | 当MDEntryType = v | IOPV |
| 16 | AuctionPrice | MDEntryPx | double | uint64, N13(5) | 8 | 111 | 当MDEntryType = w | 昨IOPV |
| 17 | AuctionQty | MDEntryPx | double | uint64, N13(5) | 8 | 119 | 当MDEntryType = x | 动态参考价 |
| 18 | Opt ionPosVol ume | MDEntrySize | longlong64 | uint64, N12 | 8 | 127 | 当MDEntryType = x | 虚拟匹配数量 |
| 19 | TradingPhaseCode | MDEntrySize | longlong64 | uint64, N12 | 8 | 135 | 当MDEntryType = z2 | 总持仓量 |
| 20 | TradingPhaseCode | TradingPhaseCode | char[9] | char[8] | 9 | 143 | 左对齐，前8位无定义为空格，第9位为0。每一位的含义详见5.1.2第4条《产品所处的交易阶段代码说明表-SSE》 | 产品所处的交易阶段代码 |
| 21 | BidPrice[0] | MDEntryPx | double | uint64, N13(5) | 8 | 152 | 当MDEntryType = 0, MDEntryPositionNo=0 | 买一价 |
| 22 | BidPrice[1] | MDEntryPx | double | uint64, N13(5) | 8 | 160 | 当MDEntryType = 0, MDEntryPositionNo=1 | 买二价 |

| 编号 | TIMD字段 | 交易所字段 | TIMD类型 | 交易所类型 | 长度 | 偏移量 | 字段详情 | 备注 |
|---|---|---|---|---|---|---|---|---|
| 23 | BidPrice[2] | MDEntryPx | double | uint64, N13(5) | 8 | 168 | 当MDEntryType = 0, MDEntryPositionNo=2 | 买三价 |
| 24 | BidPrice[3] | MDEntryPx | double | uint64, N13(5) | 8 | 176 | 当MDEntryType = 0, MDEntryPositionNo=3 | 买四价 |
| 25 | BidPrice[4] | MDEntryPx | double | uint64, N13(5) | 8 | 184 | 当MDEntryType = 0, MDEntryPositionNo=4 | 买五价 |
| 26 | BidVolume[0] | MDEntrySize | double | uint64, N12 | 8 | 192 | 当MDEntryType = 0, MDEntryPositionNo=0 | 买一量 |
| 27 | BidVolume[1] | MDEntrySize | double | uint64, N12 | 8 | 200 | 当MDEntryType = 0, MDEntryPositionNo=1 | 买二量 |
| 28 | BidVolume[2] | MDEntrySize | double | uint64, N12 | 8 | 208 | 当MDEntryType = 0, MDEntryPositionNo=2 | 买三量 |
| 29 | BidVolume[3] | MDEntrySize | double | uint64, N12 | 8 | 216 | 当MDEntryType = 0, MDEntryPositionNo=3 | 买四量 |
| 30 | BidVolume[4] | MDEntrySize | double | uint64, N12 | 8 | 224 | 当MDEntryType = 0, MDEntryPositionNo=4 | 买五量 |
| 31 | AskPrice[0] | MDEntryPx | double | uint64, N13(5) | 8 | 232 | 当MDEntryType = 1, MDEntryPositionNo=0 | 卖一价 |
| 32 | AskPrice[1] | MDEntryPx | double | uint64, N13(5) | 8 | 240 | 当MDEntryType = 1, MDEntryPositionNo=1 | 卖二价 |
| 33 | AskPrice[2] | MDEntryPx | double | uint64, N13(5) | 8 | 248 | 当MDEntryType = 1, MDEntryPositionNo=2 | 卖三价 |
| 34 | AskPrice[3] | MDEntryPx | double | uint64, N13(5) | 8 | 256 | 当MDEntryType = 1, MDEntryPositionNo=3 | 卖四价 |
| 35 | AskPrice[4] | MDEntryPx | double | uint64, N13(5) | 8 | 264 | 当MDEntryType = 1, MDEntryPositionNo=4 | 卖五价 |
| 36 | AskVolume[0] | MDEntrySize | double | uint64, N12 | 8 | 272 | 当MDEntryType = 1, MDEntryPositionNo=0 | 卖一量 |
| 37 | AskVolume[1] | MDEntrySize | double | uint64, N12 | 8 | 280 | 当MDEntryType = 1, MDEntryPositionNo=1 | 卖二量 |
| 38 | AskVolume[2] | MDEntrySize | double | uint64, N12 | 8 | 288 | 当MDEntryType = 1, MDEntryPositionNo=2 | 卖三量 |
| 39 | AskVolume[3] | MDEntrySize | double | uint64, N12 | 8 | 296 | 当MDEntryType = 1, MDEntryPositionNo=3 | 卖四量 |
| 40 | AskVolume[4] | MDEntrySize | double | uint64, N12 | 8 | 304 | 当MDEntryType = 1, MDEntryPositionNo=4 | 卖五量 |

### 5.1.2 字段说明

1. 字段无意义或无该字段行情数据除特殊说明外，字符填空格，数值填 0。

2. 交易所字段类型Nx(y)代表最大长度为 x 位数字，其中最末 y 位数字为小数部分，TIMD已对此类型的数据处理，以浮点类型发出。

3. 对**数量单位说明**如下：

   a) 产品价格、金额单位，除 B 股为美元外，其他为人民币元；

   b) 指数的成交数量(TotalVolumeTrade)为参与计算相应指数的交易数量，股票指数交易数量单位是 100股，基金指数的交易数量单位是 100 份，债券指数的交易数量单位是手；指数成交金额(TotalValueTrade)为参与计算相应指数的成交金额；

   c) 各类产品数量与成交数量单位，股票为股，基金、公募 REITs 为份，债券与回购为手，期权合约的数量单位为张；

   d) 当产品代码为债券（国债、企债、可转债）时，由于债券交易的数量以手为单位，成交金额为该债券每笔成交的价格*数量*10 的总和；当产品代码为席位质押式国债回购代码 201***、席位质押式企业债回购代码 202***以及账户质押式国债回购代码 204***时，成交金额为 100*成交数量*10；当产品代码为买断式国债回购代码 203***时，成交金额为其基础产品昨日收盘价*成交数量*10。

4. 《产品所处的交易阶段代码说明表-SSE》

   实时阶段及标志(TradingPhaseCode)为 9 位字符串，左起每位表示特定的含义，前8位**无定义则填空格，第9位为0**。该字段具体含义在不同行情类别时说明如下：

MdType=1时

1 = 指数类产品

全为空格（预留）

当MdType=2，3，4时TradingPhaseCode对应的意义

2 = 股票（A、B 股）类产品

3 = 债券类产品

4 = 基金类产品（含公募REITs）

| 位数 | 值 | 代表意义 |
| --- | --- | --- |
| 第0位 | ‘S’ | 启动（开市前）时段 |
| | ‘C’ | 开盘集合竞价时段 |
| | ‘T’ | 连续交易时段 |
| | ‘E’ | 闭市时段 |
| | ‘P’ | 产品停牌 |
| | ‘M’ | 可恢复交易的熔断时段（盘中集合竞价） |
| | ‘N’ | 不可恢复交易的熔断时段（暂停交易至闭市） |
| | ‘U’ | 收盘集合竞价时段 |
| 第1位 | ‘0’ | 此产品不可正常交易 |
| | ‘1’ | 此产品可正常交易 |
| 第2位 | ‘0’ | 未上市 |
| | ‘1’ | 已上市 |
| 第3位 | ‘0’ | 此产品在当前时段不接受进行新订单申报 |
| | ‘1’ | 此产品在当前时段可接受进行新订单申报 |

MdType=5时，TradingPhaseCode对应的意义

5 = 国债预发行产品

| 位数 | 值 | 代表意义 |
| --- | --- | --- |
| 第0位 | ‘S’ | 启动（开市前）时段 |
| | ‘C’ | 集合竞价时段 |
| | ‘T’ | 连续交易时段 |
| | ‘E’ | 闭市时段 |
| | ‘P’ | 产品停牌 |

MdType=6时，TradingPhaseCode对应的意义

6 = 盘后固定价格

| 位数 | 值 | 代表意义 |
| --- | --- | --- |
| 第0位 | ‘I’ | 启动（开市前）时段 |
| | ‘A’ | 集中撮合时段 |
| | ‘H’ | 连续交易时段 |
| | ‘D’ | 闭市时段 |
| | ‘F’ | 停牌 |

MdType=7时，TradingPhaseCode对应的意义

7 = 期权类产品

| 位数 | 值 | 代表意义 |
| --- | --- | --- |
| 第0位 | ‘S’ | 启动（开市前）时段 |
| | ‘C’ | 集合竞价时段 |
| | ‘T’ | 连续交易时段 |
| | ‘B’ | 示休市时段 |
| | ‘E’ | 闭市时段 |
| | ‘V’ | 波动性中断 |
| | ‘P’ | 临时停牌 |
| | ‘M’ | 可恢复交易的熔断时段（盘中集合竞价） |
| | ‘N’ | 不可恢复交易的熔断时段（暂停交易至闭市） |
| | ‘U’ | 收盘集合竞价时段 |
| 第1位<br><br>预留，暂填空格 | ‘0’ | 未连续停牌 |
| | ‘1’ | 连续停牌 |
| 第2位 | ‘0’ | 不限制开仓 |
| | ‘1’ | 限制备兑开仓 |
| | ‘2’ | 限制卖出开仓 |
| | ‘3’ | 限制卖出开仓、备兑开仓 |
| | ‘4’ | 限制买入开仓 |
| | ‘5’ | 限制买入开仓、备兑开仓 |
| | ‘6’ | 限制买入开仓、卖出开仓 |
| | ‘7’ | 限制买入开仓、卖出开仓、备兑开仓 |
| 第3位<br><br>仅在交易时段有效，在非交易时段无效 | ‘0’ | 此产品在当前时段不接受进行新订单申报 |
| | ‘1’ | 此产品在当前时段可接受进行新订单申报 |

5. 《行情类目详情参考表》

| 行情类目 | 行情详情 |
| --- | --- |
| **MdType = 1**<br>1 = 指数类产品 | 昨收价、最新价、开盘价、最高价、最低价、今收价 |
| **MdType = 2，3，5**<br>2 = 股票（A、B 股）类产品<br>3 = 债券类产品<br>5 = 国债预发行产品 | 昨收价、最新价、开盘价、最高价、最低价、今收价、今结算价、昨结算价 |
| **MdType = 4**<br>4 = 基金类产品（含公募REITs） | 昨收价、最新价、开盘价、最高价、最低价、今收价、今结算价、昨结算价、IOPV、昨IOPV |
| **MdType = 6**<br>6 = 盘后固定价格 | 昨收价、今收价、买一量、卖一量<br>买一数量表示当前未成交的买入申报总股数，卖一数量表示当前未成交的卖出申报总股数。 |
| **MdType = 7**<br>7 = 期权类产品 | 昨收价、最新价、开盘价、最高价、最低价、今收价、昨结算价、动态参考价、虚拟匹配数量、总持仓量（单位张） |

## 5.2 快照行情-SZSE-L1/L2（549L）

### 5.2.1 快照字段对照表

下表为**深圳证券交易所快照行情**包含的所有与交易所原始字段的对照表格。深圳证券交易所level1和level2行情快照共享一个结构体。

| 编号 | TIMD字段 | 交易所字段 | TIMD类型 | 交易所类型 | 长度 | 偏移量 | 字段详情 | 备注 |
|---|---|---|---|---|---|---|---|---|
| 1 | MdType | MDStreamID | char | char[3] | 1 | 0 | **10 = 现货**（股票，基金，债券等）集中竞价交易快照行情<br>20 = 质押式回购交易快照行情<br>30 = 债券分销快照行情<br>40 = 期权集中竞价交易快照行情<br>60 = 以收盘价交易的盘后定价交易快照行情<br>61 = 以成交量加权平均价交易的盘后定价交易快照行情<br>63 = 港股实时行情<br>**90 = 指数快照行情**<br>91 = 成交量统计指标快照行情 | 行情类别 |
| 2 | UpdateTime | OrigTime | uint32 | Int64 | 4 | 1 | 格式为HHMMSSsss | 最新更新时间 |
| 3 | SecurityID | SecurityID | char[10] | char[8] | 10 | 5 | 左对齐，多余字符为0 | 证券代码 |
| 4 | SecurityIDSource | SecurityIDSource | char[5] | char[4] | 5 | 15 | 102 = 深圳证券交易所<br>103 = 香港交易所 | 证券代码源 |
| 5 | TradingPhaseCode | TradingPhaseCode | char[9] | char[8] | 9 | 20 | 左对齐，每一位的含义详见5.2.2第4条《产品所处的交易阶段代码说明表-SZSE》 | 产品所处的交易阶段代码 |
| 6 | PreClosePrice | PrevClosePx | double | Int64,N13(4) | 8 | 29 | | 昨收价（昨日收盘指数） |
| 7 | NumTrades | NumTrades | Longlong64 | Int64 | 8 | 37 | | 成交笔数 |
| 8 | TotalVolumeTrade | TotalVolumeTrade | double | Int64,N15(2) | 8 | 45 | | 成交总量 |
| 9 | TotalValueTrade | TotalValueTrade | double | Int64,N18(4) | 8 | 53 | | 成交总金额 |
| 10 | LastPrice | MDEntryPx | double | Int64,N18(6) | 8 | 61 | 当MDEntryType=2 | 最近成交价（当前指数） |
| 11 | OpenPrice | MDEntryPx | double | Int64,N18(6) | 8 | 69 | 当MDEntryType=4 | 开盘成交价（开盘指数） |
| 12 | HighestPrice | MDEntryPx | double | Int64,N18(6) | 8 | 77 | 当MDEntryType=7 | 最高成交价（最高指数） |
| 13 | LowestPrice | MDEntryPx | double | Int64,N18(6) | 8 | 85 | 当MDEntryType=8 | 最低成交价（最低指数） |

| 编号 | TIMD字段 | 交易所字段 | TIMD类型 | 交易所类型 | 长度 | 偏移量 | 字段详情 | 备注 |
|---|---|---|---|---|---|---|---|---|
| 14 | VolumeRate | MDEntryPx | double | Int64,N18(6) | 8 | 93 | 当MDEntryType=9 | 实时成交量加权平均利率，精确到5位小数(质押式回购产品有效) |
| 15 | UDPrice1 | MDEntryPx | double | Int64,N18(6) | 8 | 101 | 当MDEntryType=x1 | 升跌1(最近价减去昨收价) |
| 16 | UDPrice2 | MDEntryPx | double | Int64,N18(6) | 8 | 109 | 当MDEntryType=x2 | 升跌2(最近价减去上一最近价，若刚开出开盘价，则=最近价-昨收价) |
| 17 | BuyOrdAvgPrice | MDEntryPx | double | Int64,N18(6) | 8 | 117 | 当MDEntryType=x3 | 买入委托数量加权平均价 |
| 18 | BuyOrdVolume | MDEntrySize | double | Int64,N15(2) | 8 | 125 | 当MDEntryType=x3 | 买入委托总数量 |
| 19 | SellOrdAvgPrice | MDEntryPx | double | Int64,N18(6) | 8 | 133 | 当MDEntryType=x4 | 卖出委托数量加权平均价 |
| 20 | SellOrdVolume | MDEntrySize | double | Int64,N15(2) | 8 | 141 | 当MDEntryType=x4 | 卖出委托总数量 |
| 21 | PERatio1 | MDEntryPx | double | Int64,N18(6) | 8 | 149 | 当MDEntryType=x5 | 股票市盈率一(股票类产品有效) |
| 22 | PERatio2 | MDEntryPx | double | Int64,N18(6) | 8 | 157 | 当MDEntryType=x6 | 股票市盈率二(股票类产品有效) **预留条目，暂不发布** |
| 23 | FundValue | MDEntryPx | double | Int64,N18(6) | 8 | 165 | 当MDEntryType=x7 | 基金净值(基金类产品有效) |
| 24 | FundRefValue | MDEntryPx | double | Int64,N18(6) | 8 | 173 | 当MDEntryType=x8 | 基金实时参考净值 |
| 25 | WarrantPremiumRate | MDEntryPx | double | Int64,N18(6) | 8 | 181 | 当MDEntryType=x9 | 权证溢价率(权证类有效有效) |
| 26 | LimitUpPrice | MDEntryPx | double | Int64,N18(6) | 8 | 189 | 当MDEntryType=xe | 涨停价 |
| 27 | LimitDownPrice | MDEntryPx | double | Int64,N18(6) | 8 | 197 | 当MDEntryType=xf | 跌停价 |

| 编号 | TIMD字段 | 交易所字段 | TIMD类型 | 交易所类型 | 长度 | 偏移量 | 字段详情 | 备注 |
|---|---|---|---|---|---|---|---|---|
| 28 | OptionPosVolume | MDEntrySize | double | Int64,N15(2) | 8 | 205 | 当MDEntryType=xg | 期权合约的持仓量(期权类有效) |
| 29 | AvgRateBp | MDEntryPx | double | Int64,N18(6) | 8 | 213 | 当MDEntryType=xj | 质押式回购的加权 平均利率涨跌BP(质押式回购产品有效) |
| 30 | PreCloseVolAvgRate | MDEntryPx | double | Int64,N18(6) | 8 | 221 | 当MDEntryType=xk | 质押式回购的昨收盘成交量加权平均利率(质押式回购产品有效) |
| 31 | BidPrice[0]-BidPrice[9] | MDEntryPx | double*10 | Int64,N18(6) | 8*10 | 229 | 当MDEntryType=0 | 买价 |
| 32 | BidVolume[0]-BidVolume[9] | MDEntrySize | double*10 | Int64,N15(2) | 8*10 | 309 | 当MDEntryType=0 | 买量 |
| 33 | AskPrice[0]-AskPrice[9] | MDEntryPx | double*10 | Int64,N18(6) | 8*10 | 389 | 当MDEntryType=1 | 卖价 |
| 34 | AskVolume[0]-AskVolume[9] | MDEntrySize | double*10 | Int64,N15(2) | 8*10 | 469 | 当MDEntryType=1 | 卖量 |

### 5.2.2 字段说明

1. 《产品所处的交易阶段代码说明表-SZSE》

   实时阶段及标志(TradingPhaseCode)为 9 位字符串，**左起**每位表示特定的含义，前8位**无定义则填空格，第9位为0**。该字段具体含义在不同行情类别时说明如下：

| 位数 | 值 | 代表意义 |
|---|---|---|
| 第0位 | ‘S’ | 启动（开市前） |
| | ‘O’ | 开盘集合竞价 |
| | ‘T’ | 连续交易 |
| | ‘B’ | 休市 |
| | ‘C’ | 收盘集合竞价 |
| | ‘E’ | 已闭市 |
| | ‘H’ | 临时停牌 |
| | ‘A’ | 盘后交易 |
| | ‘V’ | 波动性中断 |
| 第1位 | ‘0’ | 正常状态 |
| | ‘1’ | 全天停牌 |

## 5.3 逐笔委托行情-SZSE（57L）

### 5.3.1 逐笔委托对照表

| 编号 | TIMD字段 | 交易所字段 | TIMD类型 | 交易所类型 | 长度 | 偏移量 | 字段详情 | 备注 |
|---|---|---|---|---|---|---|---|---|
| 1 | MdType | MDStreamID | char | char[3] | 1 | 0 | 11 = 现货（股票，基金，债券等）集中竞价交易逐笔行情<br>21 = 质押式回购交易逐笔行情<br>41 = 期权集中竞价交易逐笔行情<br>51 = 协议交易逐笔意向行情<br>52 = 协议交易逐笔定价行情<br>71 = 转融通证券出借逐笔行情 | 行情类别 |
| 2 | UpdateTime | TransactTime | uint32 | Int64 | 4 | 1 | 格式为HHMMSSsss | 最新更新时间 |
| 3 | SecurityID | SecurityID | char[10] | char[8] | 10 | 5 | | 证券代码 |
| 4 | SecurityIDSource | SecurityIDSource | char[5] | char[4] | 5 | 15 | | 证券代码源 |
| 5 | OrdPrice | Price | double | Int64,N13(4) | 8 | 20 | | 委托价格 |
| 6 | OrderQty | OrderQty | double | Int64,N15(2) | 8 | 28 | | 委托数量 |
| 7 | Side | Side | char | char | 1 | 36 | 1 = 买<br>2 = 卖<br>G = 借入<br>F = 出借 | 买卖方向 |
| 8 | ChannelNo | ChannelNo | uint16 | uInt16 | 2 | 37 | | 频道代码 |
| 9 | ApplSeqNum | ApplSeqNum | longlong64 | Int64 | 8 | 39 | | 消息记录号 |
| 10 | OrdType | OrdType | char | char | 1 | 47 | 1 = 市价<br>2 = 限价<br>U = 本方最优 | 订单类型 |
| 11 | ConfirmID | ConfirmID | char[9] | char[8] | 9 | 48 | | 定价行情约定号 |
| | | | | | | | | |

## 5.4 逐笔成交行情-SZSE（63L）

### 5.4.1 逐笔成交对照表

| 编号 | TIMD字段 | 交易所字段 | TIMD类型 | 交易所类型 | 长度 | 偏移量 | 字段详情 | 备注 |
|---|---|---|---|---|---|---|---|---|
| 1 | MdType | MDStreamID | char | char[3] | 1 | 0 | 11 = 现货（股票，基金，债券等）集中竞价交易逐笔行情<br>21 = 质押式回购交易逐笔行情<br>41 = 期权集中竞价交易逐笔行情<br>51 = 协议交易逐笔意向行情<br>52 = 协议交易逐笔定价行情<br>71 = 转融通证券出借逐笔行情 | 行情类别 |
| 2 | UpdateTime | TransactTime | uint32 | Int64 | 4 | 1 | HHMMSSsss | 最新更新时间 |
| 3 | SecurityID | SecurityID | char[10] | char[8] | 10 | 5 | | 证券代码 |
| 4 | SecurityIDSource | SecurityIDSource | char[5] | char[4] | 5 | 15 | | 证券代码源 |
| 5 | BidApplSeqNum | BidApplSeqNum | longlong64 | Int64 | 8 | 20 | 从 1 开始计数，0 表示无对应委托 | 买方委托索引 |
| 6 | OfferApplSeqNum | OfferApplSeqNum | longlong64 | Int64 | 8 | 28 | 从 1 开始计数，0 表示无对应委托 | 卖方委托索引 |
| 7 | LastPrice | LastPx | double | Int64, N13(4) | 8 | 36 | | 成交价格 |
| 8 | LastQty | LastQty | double | Int64, N15(2) | 8 | 44 | | 成交数量 |
| 9 | ExecType | ExecType | char | char | 1 | 52 | 4 = 撤销<br>F = 成交 | 成交类别 |
| 10 | ChannelNo | ChannelNo | uint16 | uInt16 | 2 | 53 | | 频道代码 |
| 11 | ApplSeqNum | ApplSeqNum | longlong64 | Int64 | 8 | 55 | 从 1 开始计数 | 消息记录号 |
| | | | | | | | | |

# 六、DEMO

## 6.1 行情结构体概要

TIMD现支持5种组播行情的接收，分别为：

- 上海证券交易所L1行情快照
- 深圳证券交易所L1行情快照
- 深圳证券交易所L2行情快照
- 深圳证券交易所逐笔委托行情
- 深圳证券交易所逐笔成交行情

## 6.2 行情结构体详情

以下列出了6.1.2所述所有行情的结构体：

```
#ifndef __TI_MD_DATA_STRUCT_PUB_H__
#define __TI_MD_DATA_STRUCT_PUB_H__

#pragma pack(push, 1)

//上海证券交易所(SSE)行情快照结构体(L1)
struct Sse_MarketData_L1
{
  char   MdType;                  // 行情类别
  unsigned int UpdateTime : 32;   // 最新更新时间 HHMMSSsss
  char SecurityID[10];            // 证券代码
  double PreClosePrice;           // 昨收价
  long long NumTrades : 64;       // 成交笔数
  double TotalVolumeTrade;        // 成交数量
  double TotalValueTrade;         // 成交总金额
  double LastPrice;               // 最新价
  double OpenPrice;               // 开盘价
  double HighestPrice;            // 最高价
  double LowestPrice;             // 最低价
  double ClosePrice;              // 今收价
  double SettlePrice;             // 今结算价
  double PreSettlePrice;          // 昨结算价
  double Iopv;                    // IOPV
```

```
  double PreIopv;                 // 昨IOPV
  double AuctionPrice;            // 动态参考价
  long long AuctionQty:64;        // 虚拟匹配数量
  long long OptionPosVolume : 64; // 总持仓量
  char TradingPhaseCode[9];       // 产品所处的交易阶段代码
  double BidPrice[5];
  double BidVolume[5];
  double AskPrice[5];
  double AskVolume[5];
};
const int Sse_MarketData_L1_s = sizeof(Sse_MarketData_L1);

//深圳证券交易所(SZSE)行情快照结构体(L2和L1共享一个结构)
struct Szse_MarketData
{
  char   MdType;                  // 行情类别
  unsigned int UpdateTime : 32;   // 最新更新时间 HHMMSSsss
  char SecurityID[10];            // 证券代码
  char SecurityIDSource[5];       // 证券代码源. 102=深圳证券交易所. 103=香港交易所
  char TradingPhaseCode[9];       // 产品所处的交易阶段代码
  double PreClosePrice;           // 昨收价
  long long NumTrades : 64;       // 成交笔数(成交量)
  double TotalVolumeTrade;        // 成交总量
  double TotalValueTrade;         // 成交总金额
  double LastPrice;               // 最近成交价
  double OpenPrice;               // 开盘成交价
  double HighestPrice;            // 最高成交价
  double LowestPrice;             // 最低成交价
  double VolumeRate;              // 实时成交量加权平均利率，精确到5位小数(质押式回购产品有效)
  double UDPrice1;                // 升跌1(最近价减去昨收价)
  double UDPrice2;                // 升跌2(最近价减去上一最近价，若刚开出开盘价，则=最近价-昨收价)
  double BuyOrdAvgPrice;          // 买入委托数量加权平均价
  double BuyOrdVolume;            // 买入委托总数量
  double SellOrdAvgPrice;         // 卖出委托数量加权平均价
  double SellOrdVolume;           // 卖出委托总数量
  double PERatio1;                // 股票市盈率一(股票类产品有效)
  double PERatio2;                // 股票市盈率二(股票类产品有效)
  double FundValue;               // 基金净值(基金类产品有效)
  double FundRefValue;            // 基金实时参考净值
  double WarrantPremiumRate;      // 权证溢价率(权证类有效有效)
  double LimitUpPrice;            // 涨停价
  double LimitDownPrice;          // 跌停价
  double OptionPosVolume;         // 期权合约的持仓量(期权类有效)
  double AvgRateBp;               // 质押式回购的加权平均利率涨跌BP(质押式回购产品有效)
  double PreCloseVolAvgRate;      // 质押式回购的昨收盘成交量加权平均利率(质押式回购产品有效)
  double BidPrice[10];
  double BidVolume[10];
  double AskPrice[10];
  double AskVolume[10];
};
const int Szse_MarketData_s = sizeof(Szse_MarketData);

//深圳证券交易所(SZSE)逐笔委托行情
struct Szse_OrderMD
{
  char   MdType;                  // 行情类别
  unsigned int UpdateTime : 32;   // 最新更新时间 HHMMSSsss
  char   SecurityID[10];          // 证券代码
  char   SecurityIDSource[5];     // 证券代码源. 102=深圳证券交易所; 103=香港交易所
  double OrdPrice;                // 委托价格
  double OrderQty;                // 委托数量
  char   Side;                    // 买卖方向. 1=买; 2=卖; G=借入; F=出借
  unsigned short ChannelNo : 16;  // 频道代码
  long long ApplSeqNum : 64;      // 消息记录号
  char      OrdType;              // 订单类型. 1=市价; 2=限价; U=本方最优
  char      ConfirmID[9];         // 定价行情约定号
};
const int Szse_OrderMD_s = sizeof(Szse_OrderMD);

//深圳证券交易所(SZSE)逐笔成交行情
struct Szse_TradeMD
{
  char   MdType;                  // 行情类别
  unsigned int UpdateTime : 32;   // 最新更新时间 HHMMSSsss
  char         SecurityID[10];    // 证券代码
  char         SecurityIDSource[5]; // 证券代码源. 102=深圳证券交易所; 103=香港交易所
  long long    BidApplSeqNum : 64;  // 买方委托索引
  long long    OfferApplSeqNum : 64; // 卖方委托索引
  double       LastPrice;         // 成交价格
  double       LastQty;           // 成交数量
  char         ExecType;          // 成交类别. 4=撤销; F=成交
  unsigned short ChannelNo : 16;  // 频道代码
  long long    ApplSeqNum : 64;   // 消息记录号
};
const int Szse_TradeMD_s = sizeof(Szse_TradeMD);
```

```
#pragma pack(pop)
#endif
```

## 6.3 接收行情demo

TIMD无需库直接利用socket加入组播组，就可以接收转发的行情。您也可以根据需要使用各类低延迟网卡的硬件接口接收TIMD行情，以最快接收行情。

**以下demo是以上海证券交易所(SSE)一档行情快照结构体（Sse_MarketData_L1）为例子**，接收SSE level1行情的示列代码。

如果您需要接收其他类型的行情，请您修改对应行情的**组播网络地址**和**所需行情结构体**。

您也可以参考附件中的demo程序。

```
#include <iostream>
#include <cstdio>
#include <cstdlib>
#include <string.h>
#ifdef WIN32
#include <winsock.h>
#else
#include <arpa/inet.h>
#include "sys/socket.h"
#include "netinet/in.h"
#include "unistd.h"
#include "net/if.h"
#endif

#include "timddatastructpub.h"

//组播网络IP地址
#define MULTIIP "239.20.30.90"
//组播网络端口号
#define MULTIPORT 10908
//接收组播数据的本地网卡ip地址
#define LOCALIP "127.0.0.1"

int main()
{
  int ret;
  struct sockaddr_in    addrMulti;  //组播地址
  struct sockaddr*    addrPtr;      //组播地址指针
  unsigned int      addrLen;        //组播地址长度
  struct ip_mreq      mReq;

  //建立socket
  int mysocket=socket(AF_INET,SOCK_DGRAM,0);
  if(mysocket<0)
  {
    printf("建立socket失败\n");
    return -2;
  }

  //允许程序的多个实例运行在同一台机器上
  int on = 1;
  if(setsockopt(mysocket,SOL_SOCKET,SO_REUSEADDR,&on,sizeof(on))<0)
  {
    printf("允许程序的多个实例运行在同一台机器上失败\n");
    return -3;
  }

  //组播地址赋值
  addrMulti.sin_family=AF_INET;
  addrMulti.sin_port=htons(MULTIPORT);
  addrMulti.sin_addr.s_addr=inet_addr(MULTIIP);

  //绑定地址结构到套接字
  ret=bind(mysocket,(struct sockaddr*)&addrMulti,sizeof(addrMulti));
  if(ret<0)
  {
    printf("绑定组播网络地址失败\n");
    return -4;
  }

  //设置接收超时时间为1秒
  struct timeval timeout = {1,0};
  if(setsockopt(mysocket, SOL_SOCKET, SO_RCVTIMEO,&timeout,sizeof(struct timeval)) < 0)
  {
    printf("设置组播网络超时时间失败\n");
```

```
  return -5;
}

//客户端加入组播组
mReq.imr_multiaddr.s_addr=inet_addr(MULTIIP);//组播组的IP
mReq.imr_interface.s_addr=inet_addr(LOCALIP);
ret = setsockopt(mysocket,IPPROTO_IP,IP_ADD_MEMBERSHIP,&mReq,sizeof(mReq));
if(ret<0)
{
  printf("客户端加入组播组失败\n");
  return -6;
}
else
{
  printf("客户端加入组播组:%s:%d|%s\n",MULTIIP,MULTIPORT,LOCALIP);
}

addrPtr = (struct sockaddr*)&addrMulti;
addrLen = sizeof(addrMulti);

char buffer[1024] = {0};
int len = 1024;
while(1)
{
  //接收组播行情数据
  int sz = recvfrom(mysocket,buffer,len,0,addrPtr,&addrLen);
  if(sz > 0)
  {
    if(Sse_MarketData_L1_s == sz)
    {
      Sse_MarketData_L1 *mdPtr = (Sse_MarketData_L1*)buffer;
      int wtype = mdPtr->MdType;

      std::cout << wtype << ","
                << mdPtr->UpdateTime        << "," << mdPtr->SecurityID      << ","
                << mdPtr->PreClosePrice     << "," << mdPtr->NumTrades       << ","
                << mdPtr->TotalVolumeTrade  << "," << mdPtr->TotalValueTrade << ","
                << mdPtr->LastPrice         << "," << mdPtr->OpenPrice       << ","
                << mdPtr->HighestPrice      << "," << mdPtr->LowestPrice     << ","
                << mdPtr->ClosePrice        << "," << mdPtr->SettlePrice     << ","
                << mdPtr->PreSettlePrice    << "," << mdPtr->Iopv            <<","
                << mdPtr->PreIopv           << "," << mdPtr->AuctionPrice    << ","
                << mdPtr->AuctionQty        << "," << mdPtr->OptionPosVolume << ","
                << mdPtr->TradingPhaseCode  << ","
                << mdPtr->BidPrice[0] <<"," <<mdPtr->BidVolume[0] <<","
                << mdPtr->AskPrice[0] <<"," <<mdPtr->AskVolume[0] <<","
                << mdPtr->BidPrice[1] <<"," <<mdPtr->BidVolume[1] <<","
                << mdPtr->AskPrice[1] <<"," <<mdPtr->AskVolume[1] <<","
                << mdPtr->BidPrice[2] <<"," <<mdPtr->BidVolume[2] <<","
                << mdPtr->AskPrice[2] <<"," <<mdPtr->AskVolume[2] <<","
                << mdPtr->BidPrice[3] <<"," <<mdPtr->BidVolume[3] <<","
                << mdPtr->AskPrice[3] <<"," <<mdPtr->AskVolume[3] <<","
                << mdPtr->BidPrice[4] <<"," <<mdPtr->BidVolume[4] <<","
                << mdPtr->AskPrice[4] <<"," <<mdPtr->AskVolume[4] << std::endl;
    }
  }
}

  //离开组播组
  setsockopt(mysocket,IPPROTO_IP,IP_DROP_MEMBERSHIP,&mReq,sizeof(mReq));
  close(mysocket);
}
```