Electronic Trading for Programmers

Part 1: Low-latency Execution

Portofino Technologies

Mathias Gaunard

Quantitative Development Lead

16th of January 2023

Outline

- Electronic Trading
  - Trading
  - Matching
- Low-Latency
  - Why things take time
  - Why being fast matters
  - Measuring
- Exchange connectivity
  - Feeds
  - Protocol considerations
- Trading system architecture
  - Constraints
  - Networking
  - State of the union
- Part II sneak peek

Outline

- Electronic Trading
    - Trading
    - Matching

- Low-Latency
    - Why things take time
    - Why being fast matters
    - Measuring

- Exchange connectivity
    - Feeds
    - Protocol considerations

- Trading system architecture
    - Constraints
    - Networking
    - State of the union

- Part II sneak peek

What is trading?

Exchanging assets with another party

- base/quote, often quote is a stable currency, e.g BTC/USD
- buy/sell base (sell/buy quote)
- usually done either as an investment or as speculation

Different ecosystems

- equities (stocks, indices on stocks, ETFs)
- fixed income (government bonds, corporate bonds)
- commodities (oil, metals, grain)
- currencies
- cryptocurrencies

How can I trade?

Financial instruments

- Trade assets outright (spot)
- Obtain a loan and trade against that loan (spot with margin account)
- Enter a contract with trade obligations at term (futures, perp swaps, CFD)
- Enter a contract with trade optionality at term (options, warrants)
- Smart contracts (blockchain-based enforcement)
- Exotic contracts (sophisticated legally-binding agreements)

Different products

- Listed on public exchanges
- Broker-dealer products
- Over-the-Counter only

Why trade?

Price move prediction (alpha)

- fundamental analysis of product (long term)
- events, news (medium term)
- market trends, statistics etc. (short term)

Connecting people

- arbitrage buy/sell flow
- collecting fees
- arbitrage different marketplaces
- arbitrage derivative instruments on same assets

Why electronically?

Larger pool of participants

- link venues across the world
- connect retail and professionals
- more competition, better prices

Transparency

- Records of all transactions
- Enforcement of due process
- MIFID compliance

Automation

- Enables looking at small opportunities a human wouldn’t consider
- Systematic algorithms to run strategies consistently
- Low-touch enables higher volume

Who’s trading?

Investors, buy-side

- Pension funds, mutual funds
- Venture capitalists
- Hedge funds
- Proprietary trading firms
- Retail

Trading services, sell-side

- Exchanges
- Market-makers
- Investment banks
- Brokers

How to interact

Direct, Over-the-Counter

- voice
- electronic, Request-for-Quote

Through marketplace/exchange

- best participant selected (usually anonymous)
- small fees
- multiple platforms
  - □ continuous, ”the screen”
  - □ auctions
  - □ multi-participant OTC-type platforms

Execution on-behalf

- finds best way to enter large positions over longer time periods
- larger fees
- methodology pre-agreed and/or performance-tracked
  - □ Flow traders
  - □ Algo-driven, vwap/twap

Outline

- Electronic Trading
  - Trading
  - Matching
- Low-Latency
  - Why things take time
  - Why being fast matters
  - Measuring
- Exchange connectivity
  - Feeds
  - Protocol considerations
- Trading system architecture
  - Constraints
  - Networking
  - State of the union
- Part II sneak peek

Vocabulary

- **bid**: buy order
- **ask**: sell order
- **offer**: ask
- **side**: whether it's buy or sell
- **tick**: increment of prices that are valid to place orders on
- prices p_1 **better** than p_2: $p_1 > p_2$ if bid, $p_1 < p_2$ if ask
- **BBO**: best bid and ask
- bid-ask **spread**: $best\_ask_{price} - best\_bid_{price}$
- bid and ask orders are **crossed**: $bid_{price} >= ask_{price}$
- **liquidity**: quantity addressable for trading, implied at good prices
- **touch**: liquidity close to the BBO

Continuous matching

Continuous

- Buyer/seller gets matched with sellers/buyers immediately if possible
  - □ If triggering match called **aggressor**, **taker** or **active** order
  - □ Removes matched liquidity, involved parties have traded
- Otherwise stays in order book and becomes **resting**, **maker** or **passive**
- Order book is always uncrossed
- Participants can amend/cancel their open orders

Limit orders

- Instrument identifier and side
- Maximum quantity (number of lots)
- Worst price per lot

Orders flags

- Immediate-or-Cancel
- Book-or-Cancel
- Icebergs

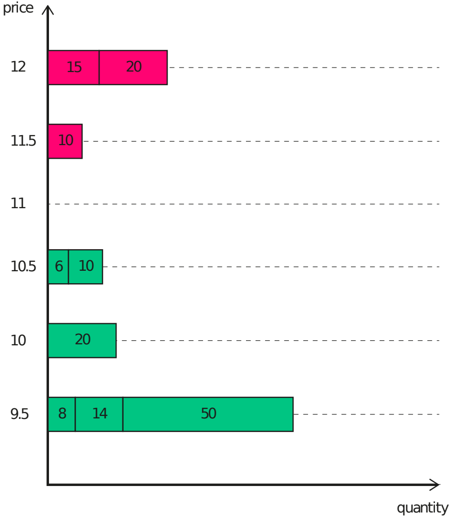
Order book, initial state

Order book

- Steady-state, all bids strictly less than asks
- Multiple orders per price level, arranged per insertion order (priority)

Example scenario

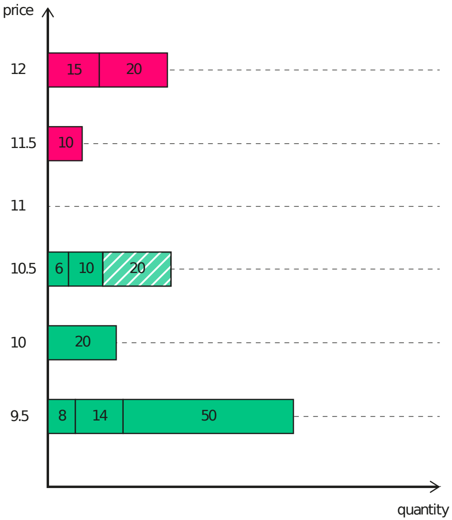
- Tick of 0.50
- Spread of 1, e.g. two ticks



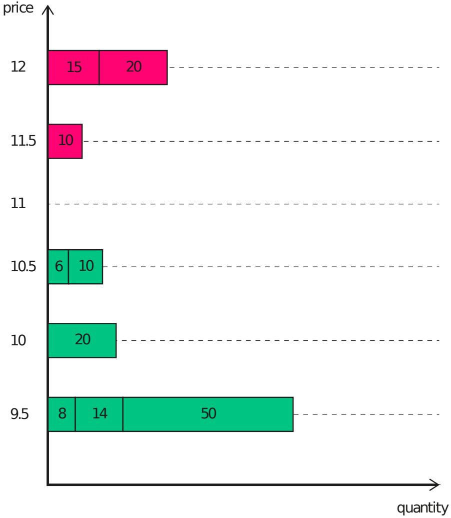
Insertion example, join

Buy 20@10.50

- Join the queue on best bid
- Spread unaffected, two ticks wide



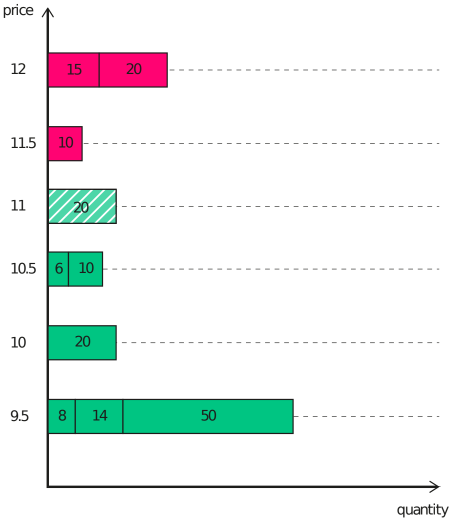
Order book, back to initial



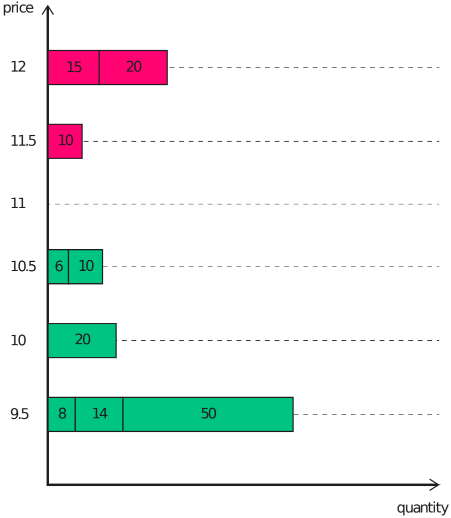
Insertion example, improve

Buy 20@11

- Establish new price level
- Front of the queue
- Spread tightened to one tick



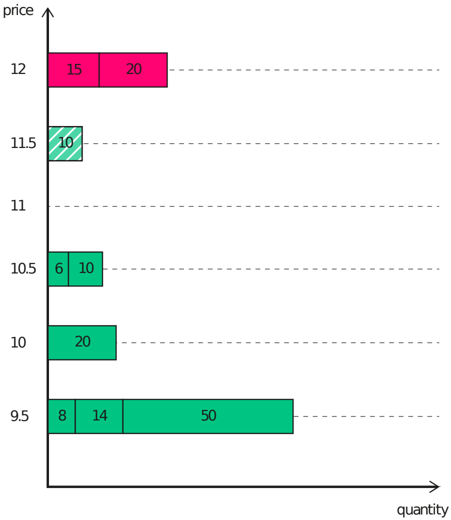
Order book, back to initial



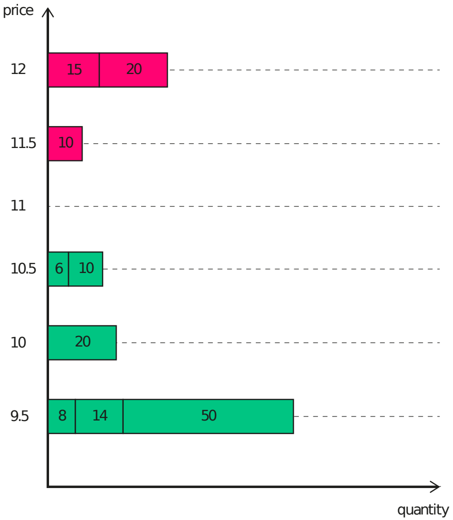
Insertion example, take and improve

Buy 20@11.50

- Trade 10@11.50
- Sell order disappears
- Establish new price level at 11.50 with remaining quantity
- Front of queue
- Spread unaffected, but market "ticked up"



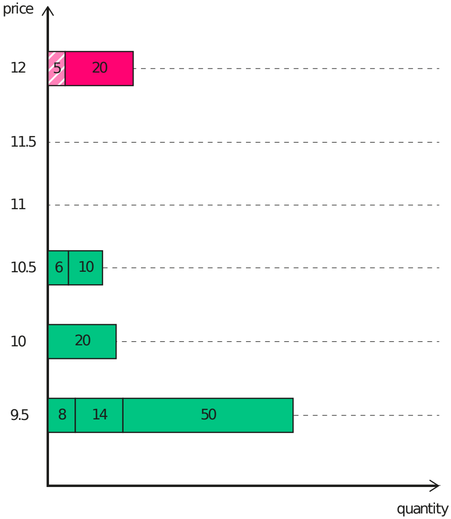
Order book, back to initial



Insertion example, take and widen

Buy 20@12

- Trade 10@11.50
- Sell order disappears
- Trade 10@12
- Buy order fully filled, not entering the book
- Sell order partially filled
- Spread widened, two ticks wide



Order book, data structures

Index orders by identifier

- Support modify and cancel
- Hash table

Track priority of orders

- Linked-list of orders per level
- Ordered sequence of levels
  - □ Self-balancing binary tree
  - □ Circular buffer, dense tick representation

Many updates per second

- Pre-allocate and pool
- Hybrid data structures
- Optimize for operations close to touch
- Optimize hashing for indexing method of exchange

Outline

- Electronic Trading
  - Trading
  - Matching

- Low-Latency
  - Why things take time
  - Why being fast matters
  - Measuring

- Exchange connectivity
  - Feeds
  - Protocol considerations

- Trading system architecture
  - Constraints
  - Networking
  - State of the union

- Part II sneak peek

Outline

Electronic Trading
    Trading
    Matching

Low-Latency
    Why things take time
    Why being fast matters
    Measuring

Exchange connectivity
    Feeds
    Protocol considerations

Trading system architecture
    Constraints
    Networking
    State of the union

Part II sneak peek

Asynchronous change requests

Constant stream of updates

- Many participants
- Orders being added/modified/cancelled multiple times per millisecond
- Exchange has to distribute a lot of data, may be slow
- May even be throttled on its way to you

Join the queue

- Observe market in a given state, want to affect a change
- Change request submitted and queued
- Exchange drains the queue of changes, eventually processes it
- Observe market with your change applied, tens of milliseconds might have passed

Fairness

Access to information

- Preferential access
- Does network provide information to all participants at the same time

Order processing ordering

- Does everyone go through the same gateway
- Is there any reordering on the way to this gateway or internally

Special rules make speed less of a concern

- Micro-auctions
- Asymmetric delays
- Pro-rata matching

Determinism

Deterministic

- Fastest always wins
- Clear, fair, efficient
- Technologically more challenging for exchange
- Leads to people building FPGAs and ASICs

Non-deterministic

- Fastest only has an advantage, but semi-random
- Exchange can have dynamic behaviour based on load
- Leads to reverse-engineering and finding whatever can be gamified to improve the odds
- Mostly lots of tricks but no need for hardware

Outline

- Electronic Trading
  - Trading
  - Matching
- Low-Latency
  - Why things take time
  - Why being fast matters
  - Measuring
- Exchange connectivity
  - Feeds
  - Protocol considerations
- Trading system architecture
  - Constraints
  - Networking
  - State of the union
- Part II sneak peek

Pick-offs

Theoretical price-driven

- Only willing to buy for less/sell for more than some price model
- Market conditions change, theo changes, you cancel
- Aggressor is faster, bad trade

Risk-driven

- Have lots of orders on many instruments or different venues
- Don’t necessarily want them to be filled all at once
- One order is filled, cancel other ones
- Aggressor is faster, overtrade

Achieve good priority

Bad priority ⇒ never get to trade

Price improvement opportunities

- Event-driven and competitive
- Tied to price move predictions

Proactively place orders where price might move to

- More load and complexity
- Additional risk in the book

Everyone wants to take the good stuff

New order crosses theo

- Spread tightened, new price attractive to your theo
- Try and send an order to match against it
- Other participants might do the same, fastest takes it

Event causes theo change

- Something happens that affects your price model (usually other market)
- Now you think many existing orders are mispriced
- Try and send orders to match against all of them
- Other participants might do the same, may only get some of it

Some metrics

Fill rate

- How often do I get filled before cancelling
- Relative to various priority metrics

Hitting rate

- How often do I get anything when I try to take liquidity
- Amount of credit/edge
- Latency of sending the message through

Outline

Electronic Trading
- Trading
- Matching

Low-Latency
- Why things take time
- Why being fast matters
- Measuring

Exchange connectivity
- Feeds
- Protocol considerations

Trading system architecture
- Constraints
- Networking
- State of the union

Part II sneak peek

Time on the wire

Trigger to order

- When was network packet triggering decision received
- When was network packet with order emitted out

Streaming at 10Gbps

- MTU of 1500 is 1200ns
- End of Frame to End of Frame
- Start of Frame to Start of Frame
- Start of Trigger to Start of Frame

Software times

Not whole picture

- Delay between packet received and picked up by software
- Delay between sending packet and it being serialized out

Low-latency measurements

- `rdtsc(p)`
- `rdpmc`

Meaningful statistics

Quantiles

- Min – how fast you can expect to get
- Median – how fast you are typically
- 90^{th} percentile – how well are you protected
- 99^{th} percentile – how bad can it get

Determinism

- Removing tails is hard, but important
- Important events happen rarely
- Control flow divergence increases jitter

Outline

Electronic Trading
- Trading
- Matching

Low-Latency
- Why things take time
- Why being fast matters
- Measuring

Exchange connectivity
- Feeds
- Protocol considerations

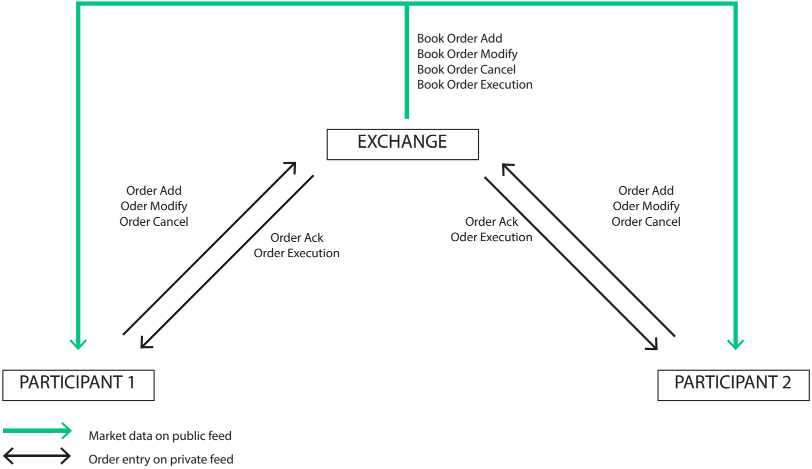
Trading system architecture
- Constraints
- Networking
- State of the union

Part II sneak peek

Outline

- Electronic Trading
  - Trading
  - Matching
- Low-Latency
  - Why things take time
  - Why being fast matters
  - Measuring
- Exchange connectivity
  - Feeds
  - Protocol considerations
- Trading system architecture
  - Constraints
  - Networking
  - State of the union
- Part II sneak peek

Participant connections



EXCHANGE

Book Order Add
Book Order Modify
Book Order Cancel
Book Order Execution

Order Add
Oder Modify
Order Cancel

Order Ack
Order Execution

Order Ack
Oder Execution

Order Add
Oder Modify
Order Cancel

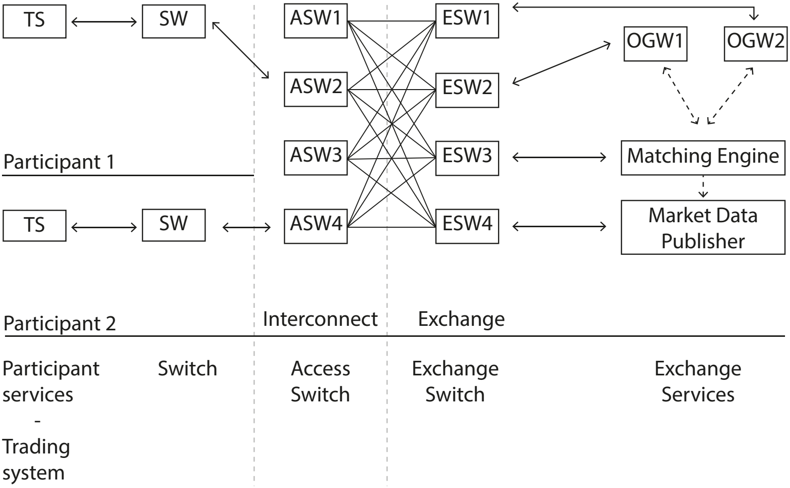
PARTICIPANT 1

PARTICIPANT 2

Market data on public feed

Order entry on private feed

Gateways and matching engine



Unicast vs Multicast

Unicast

- Send data for every single participant, bandwidth-hungry
- Goes through any router, including the open Internet
- No one gets it at the same time
- Can use TCP and have tailored per-participant data

Multicast

- Send data once, switches fan-out, bandwidth-efficient
  - □ Requires ability to propagate subscribers through network
- Participants get data at the same time
  - □ Modulo network congestion and ethernet signal phase
- UDP-only

Outline

Electronic Trading
- Trading
- Matching

Low-Latency
- Why things take time
- Why being fast matters
- Measuring

Exchange connectivity
- Feeds
- Protocol considerations

Trading system architecture
- Constraints
- Networking
- State of the union

Part II sneak peek

Order book fidelity

Levels

- Level 1: BBO
- Level 2: aggregated quantity per price level
- Level 3: all individual orders

Netting

- Unnetted
- Throttled
- Coalesced

Recovery and reliability

Sequence numbers

- Out of order
- Gaps
- Incremental updates

Recovery

- Replay since beginning
- Snapshot

Serialization formats

Binary

- Flat, `reinterpret_cast`-friendly formats, e.g. SBE
- Delta encoding, e.g. FAST

Text

- FIX, key/value pairs
- JSON

Fragmentation

- Nice exchanges avoid it
- Others whatever IP says goes

Decimal numbers

Problem

- Often working with non-integral prices and quantities
- 0.1 cannot be represented exactly with binary floating-point
  Find $\{s, m, \mathrm{e}\}$ such that $v \simeq (-1)^s \times m \times 2^{\mathrm{e}}$, with $1 \leq m < 2$
- Approximations cause all sorts of problems

Decimal floating-point

- Find $\{s, m, \mathrm{e}\}$ such that $v \simeq (-1)^s \times m \times 10^{\mathrm{e}}$, with $1 \leq m < 10$
- IEEE754-2008, decimal32, decimal64, decimal128, backed by IBM and Intel
- Hardware support in POWER, software libraries
- Remains esoteric and slow

Decimal numbers (2)

Decimal fixed-point

- Fix the exponent and denormalize the mantissa
- Straightforward implementation, scaled integers
- Beware of operations that would change the scale
- No dynamic range/precision trade-off, beware of overflows

Fixing approaches

- Compile-time exponent, part of type
- Runtime exponent, schema that applies to a dataset
  - □ no redundant storage with all values
  - □ e.g. time series

Decimal numbers in practice

Exchanges

- Some use decimal floating-point (rare)
- Most use decimal fixed-point with negative exponent
    - □ Either same exponent for everything on protocol
    - □ Or per-instrument exponent
- Some just use decimal text

Recommendations

- Store data as integers, keep track of the relevant exponent they use
- Stick to scale-preserving operations when doing exact computations
- Switch in-and-out of `double` whenever doing non-exact numerical computations

Outline

- Electronic Trading
    - Trading
    - Matching
- Low-Latency
    - Why things take time
    - Why being fast matters
    - Measuring
- Exchange connectivity
    - Feeds
    - Protocol considerations
- Trading system architecture
    - Constraints
    - Networking
    - State of the union
- Part II sneak peek

Outline

Electronic Trading
- Trading
- Matching

Low-Latency
- Why things take time
- Why being fast matters
- Measuring

Exchange connectivity
- Feeds
- Protocol considerations

Trading system architecture
- Constraints
- Networking
- State of the union

Part II sneak peek

Soft real-time constraints

By order of importance

- No system calls that block non-deterministically
- Real-time scheduling guarantees and affinity pinning
- Lock-free synchronization (i.e. no blocking system calls at all)
- No memory allocations
- No system calls at all
- Wait-free synchronization
- Limited code flow divergence (i.e. deterministic execution time)

Hybrid trading systems

Special-purpose

- Super fast cancel
  - □ Better cancel more often than needed
  - □ The simpler, the easier it is to make it fast
- Fast hitting
- Slower quoting, relaxed coding constraints

Integrated

- More precise cancelling
- Enhanced capabilities for sophisticated stategies

Outline

Electronic Trading
- Trading
- Matching

Low-Latency
- Why things take time
- Why being fast matters
- Measuring

Exchange connectivity
- Feeds
- Protocol considerations

Trading system architecture
- Constraints
- Networking
- State of the union

Part II sneak peek

Ethernet, IP, TCP

TCP – byte stream

- Handshake
- Ack window
- Nagle algorithm
- Retransmissions

IP – packets

- Routing
- MTU and fragmentation

Ethernet – frames

- Bandwidth
- Signal phase

Low-latency networking

Kernel-bypass

- Direct communication with Network Interface Adapter in userland
- DMA, write-combining memory, PCI-Express
- Disable interrupts, too slow to read means dropped data

Userland TCP/IP

- Receive/send ethernet frames or frame fragments in application
- Re-implement all of TCP and IP without relying on the kernel
- Shortcuts for reliable networks

Ready-made solutions

Solarflare (now Xilinx)

- OpenOnload, BSD socket compatibility layer, high conformance
- EFVI, low-level API
- Onboard FPGA since Xilinx acquisition

Exablaze (now Cisco)

- exanic low-level API, pre-loading capabilities
- Onboard FPGA as core of the system
- exasock, BSD socket compatiblity layer, not-quite-conforming

Others

- Mellanox (now Nvidia), more targeted at HPC and Infiniband
- Myricom, other HPC pioneer, notable for a Windows API, now defunct

Standard approaches

DPDK

- Linux foundation
- Large userland framework for kernel-bypass networking
- Supports traditional NICs (Intel, etc.)
- Userland TCP implementations

`io_uring`

- Linux kernel
- paradigm shift removing system calls
- new languages (e.g. Rust) building their networking around it

Threading model

Few threads

- No reliance on OS thread scheduling
- Cooperative scheduling intra-thread

Some frameworks

- Asio, not the best fit
- Seastar, good principles

Share-nothing

- Objects local to a given thread
- Communication via lock-free queues
- Seq-locks for state sampling
  - □ recently made compatible with C++ memory model

Outline

- Electronic Trading
  - Trading
  - Matching
- Low-Latency
  - Why things take time
  - Why being fast matters
  - Measuring
- Exchange connectivity
  - Feeds
  - Protocol considerations
- Trading system architecture
  - Constraints
  - Networking
  - State of the union
- Part II sneak peek

State of the union

Race to the bottom

- Normal software $< 10ms$
- Software with real-time in mind $< 100\mu s$ – sweet spot?
- Ultra low-latency software $< 3\mu s$
- Normal FPGA solution $< 500ns$
- Ultra low-latency FPGA solution $< 50ns$
- ASIC $< 30ns$
- Above and beyond $< 10ns$

Outline

Electronic Trading
- Trading
- Matching

Low-Latency
- Why things take time
- Why being fast matters
- Measuring

Exchange connectivity
- Feeds
- Protocol considerations

Trading system architecture
- Constraints
- Networking
- State of the union

Part II sneak peek

Part II sneak peek

Pricing

- Pricing from trading activity
- Pricing derivatives

Data analysis

- Time series
- Simulation

Risk management

- Greeks
- Slippage

Questions?