Pocket-Sized
High Frequency Trader (PSHFT)

Gabriel Blanco - gab2135
Brian Bourn - bab2177
David Naveen Dhas Arthur - da2647
Suchith Vasudevan - sv2340

Background

In the world of algorithmic financial trading, speed is king. You want to be able to react as quickly as possible to sudden changes in the market in order to get the best possible value for your trades. For the past decades, financial institutions have been developing dedicated low-level processing hardware to make buying and selling decisions as quickly as possible, a process known as *High Frequency Trading*. Thanks to advances in the power and availability of *Field Programmable Arrays* (FPGA), the barrier of entry for high frequency trading is lower than ever. Using an FPGA, you still get the speed benefits of using hardware acceleration, but at a fraction of the cost of building a dedicated processor, as well as the flexibility to reprogram it to extend its functionality.

In this project, we aim to implement the core functionality of a High Frequency Trading machine, give it set upper and lower bounds for buying and selling fake stock, then test its response time and compare it to industry level HFT to see if it is feasible to make a purely FPGA “Pocket-Sized” HFT.

Goals

Our Pocket-Sized HFT should:

- Receive large volumes of trading information via ethernet
- Process and compare the information against set upper and lower bounds using multiple priority queues
- Make a decision and “send” an accurate buy or sell response if the stock price exceeds those bounds, and most importantly:
- Do all of this processing *as quickly* as the hardware allows

Milestones

1. **Input and Output:** Using Ethernet, find a way to receive trading information, and a way to send buy and/or receive signals from the FPGA. This way, we can create a test data set to send, and verify the response, allowing us to do testing on the rest of the features
2. **Processing:** Be able to receive packets of information from the financial stream and read the name of the stock as well as the price.
3. **Algorithm:** Implementation of a HFT algorithm using multiple priority queues to compare the current stock value with the upper and lower bounds.
4. **Optimization**: Test response time of the HFT, and work to optimize the speed of the machine to reduce the amount of time it takes to accurately make a buy/sell response.