Caching on FPGAs Project Proposal

Prepared by:

Maëlle Guerre

01496040

Prepared for:

Imperial College London

Department of Electrical & Electronics Engineering

EXECUTIVE SUMMARY

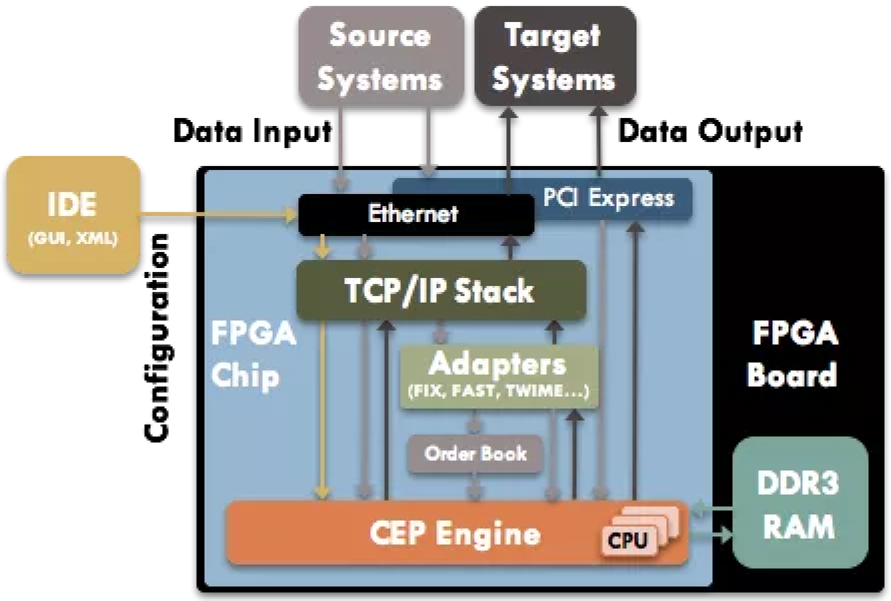
For my Final year project, I would like to work on implementing a cache coherency protocols in Verilog. After working in algorithmic trading, I came across interesting practical applications of FPGAs for speeding up TCP connections and doing fast risks checks. Having a cache like behaviour would be useful for the future of algorithmic trading as well as general learning experience. The deliverables of my master's project would be as follows:

1 **Proof of concept (PoC) of cache implementations on FPGAs**
Simulation through verilog compilation and TLA+ testing with time analysis for accessing the simulated cache.

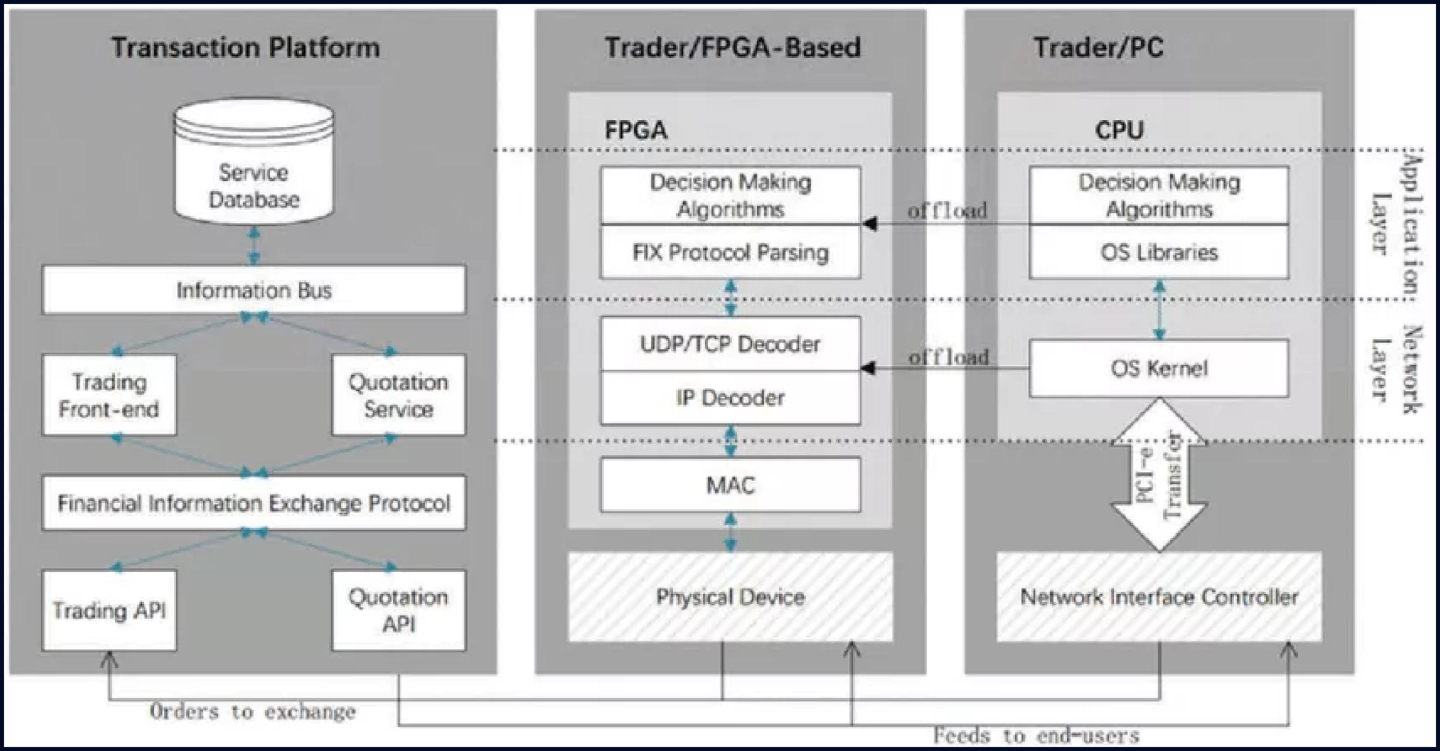
2 **Final year project (FYP) report**
Experiments, conclusions and learning throughout the process.

I have proposed this to my team at Morgan Stanley and they are willing to provide me with some support if this project works. This would be a great opportunity to do the FPGA application and deployment in a production environment after finishing my masters.
Motivation behind this project:

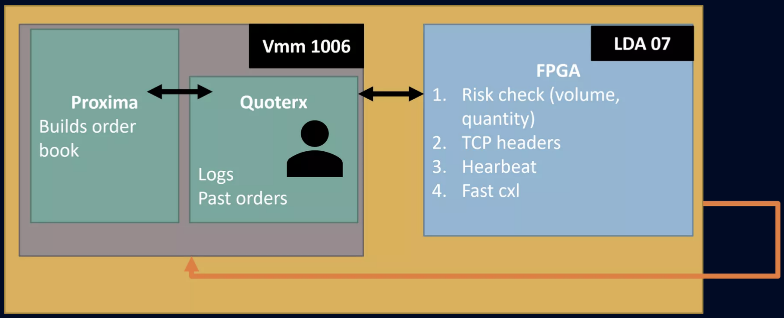
- Computer architecture, networks, Verilog and low latency study related to my degree in EIE,
  - Technical challenge - memory like behaviour is more challenging to implement than CPU like on FPGAs and has not been tested yet.
- Speeding up trades by hundreds of nanoseconds would result in major financial gains for a company using FPGA based algorithmic trading,
- Applications in Machine learning and other fields - caching models on FPGAs or hyperparameter sets during training would be interesting



ABOUT the project

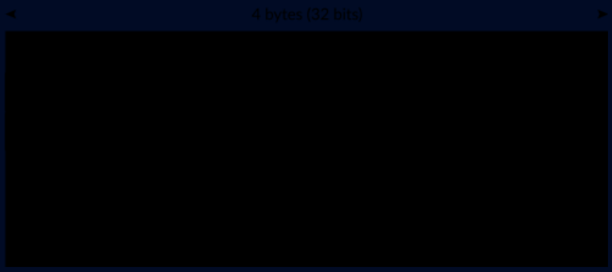


FPGAs (field programmable gate arrays) raised in popularity in Algorithmic trading for their superior speed and the ability to re-compute the lookup tables connections. Whilst software applications react in microseconds, FPGAs can cancel orders or keep a connection with the exchange in nanoseconds. Last month, I joined a team in Morgan Stanley developing and maintaining FPGAs as a gate between the trading system and the exchange.



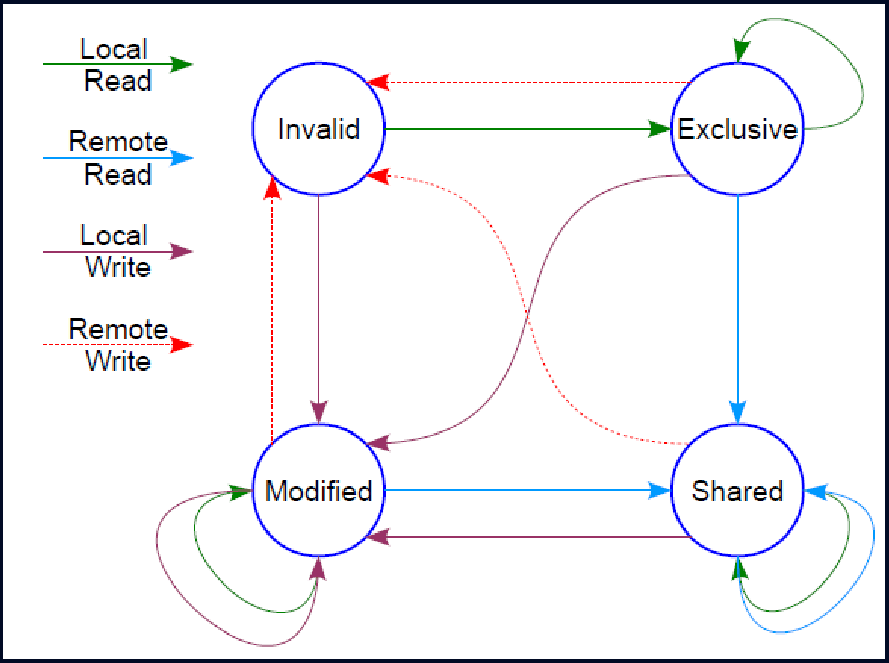
a. Order books on FPGAs
The main roles are currently:

1. Risk check: volume and quantity of the order
2. Adding corresponding TCP headers on the message,
3. 'Heartbeat' message confirmation of connection to the exchange
4. Fast cancel of an order if needed to minimise losses.

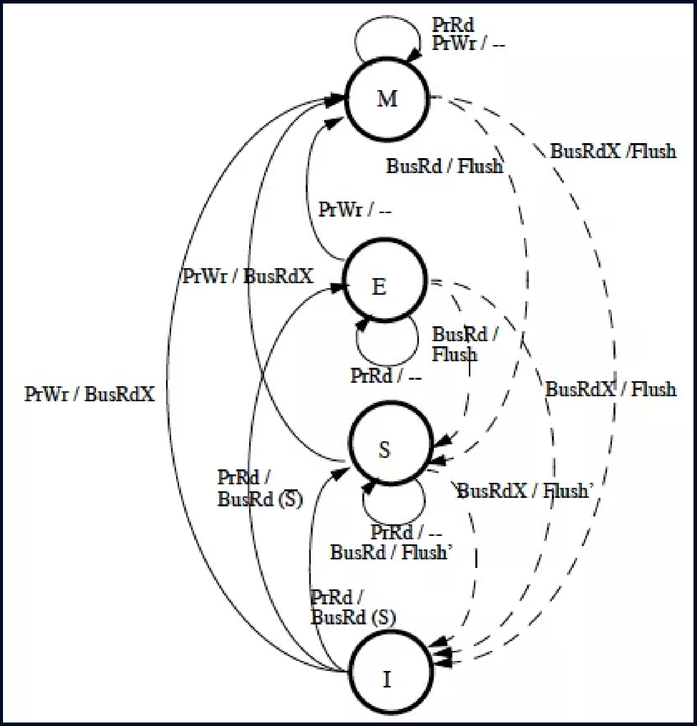


b. Building packets
This architecture lays down parts of the OSI network model on an FPGA to speedup the process of building packets and sending them on the network.

ABOUT the project



MESI protocol is an invalidate back cache coherence protocol to support write back. Since both the exchange and client are reading/writing the "memory" on the FPGA's orders, this would allow to speedup orders that are similar when repeated in a short time frame. This happens often in trading and would be a huge advantage.



The main goal of this project will be to prove that this protocol can be implemented safely on a FPGA. MESI is described as a state machine, which makes an implementation in Verilog challenging but possible.

STATEMENT OF WORK

The work will be split into the following tasks, in order to give an idea for timeframe and workload.

1. Task: Research on the project

Estimated time: 21 days

Detailed description of tasks: Project description, problem statement and collecting past research. Finding a supervisor, downloading compiling tools.

Deliverables:

1. Diagram of MESI protocol
2. Diagram of cache/memory level and snooping protocol
3. Pseudo code implementation
4. List of existing research on MESI protocol implementation
5. List of existing research on FPGA caching
6. Learning TLA+

2. Task: Verilog implementation of MESI protocol

Estimated time: 21 days

Detailed description of tasks: Write and test incrementally a cache - memory hierarchy in Verilog in order to compile the Verilog code on a FPGA.

Deliverables: Working baseline codebase that follows MESI state machine in Verilog and can be tested.

3. Task: Testbench and debugging on Verilog

Estimated time: 21 days

Detailed description of tasks: Writing testbench that access and use the caches/memory. Time and memory access analysis. Speeding up the code and making sure cache behaves as real time as possible.

Deliverables:

1. bar graph pass/fail per different tested features (cache protocols and access)
2. Time graph analysis for some testbench
3. Compiled board (virtual) power and space analysis = how much resources are used and under how much pressure they are.
4. List of improvement and codebase change, if any

4. Task: TLA+ implementation of MESI protocol

Estimated time: 21 days

Detailed description of tasks: Write the MESI protocol in TLA+ in order to allow a full test of the project. TLA+ will try all possible paths on the board and all gate combinations.

Deliverables: Graph/diagram and testcases that prove that the board is safe to use and all combinations work.

PRELIMINARY PROJECT SCHEDULE

The total period of performance of the work described in this IT Project proposal is anticipated to last (add a corresponding number) weeks. The following table provides a preliminary project schedule. This table is intended for planning purposes only.

Task Name	Duration
POC for FPGA Caching	152 days
Research on MESI, existing research & learning TLA+	14 days
Verilog implementation of MESI protocol	21 days
Testbench writing, debugging and verification (verilog)	28 days
TLA+ implementation of MESI protocol	21 days
Debugging and verification by running code (TLA+)	28 days
Diagram and drawing conclusions, reflection notes	20 days
FYP paper redaction	20 days

Use of prior knowledge

The main goal of this project is to establish a proof of concept, hence proving that it is possible to implement caching on FPGA and how to do it.
Due to time limitation, the code may not be pushed on a FPGA board directly.

This project is interesting because will make use of several skills learn during my years in Electronics and Information engineering:

1. Coding skills - good coding practices in Verilog and TLA+, C++ and pseudo code for first tests
2. Computer architecture knowledge about memory hierarchy, cache protocols and snooping.
3. Digital electronics on state machine and how to convert them to gates and code.
4. Electronic labs with the write up of tests and edge cases.
5. Analysis of circuits, Algorithm and complexity as the code will be improved to run as fast as possible on a specific board.

09 / 12 / 2021

Maëlle Guerre Imperial College London