# 國立清華大學 電機工程學系

# 實作專題研究成果摘要

# FPGA architecture for high-frequency spot market trading systems

# FPGA 架構之現貨市場高頻交易系統

專題領域：系統領域

指導教授：翁詠祿教授 (Prof. Yeong-Luh Ueng)

組員姓名：王修呈(Hsiu-Cheng Wang)、梁頊宸 (Hsu-Chen Liang)

研究期間：114 年 6 月 1 日至 114 年 11 月 24 日止，共 6 個月

# Abstract

This project presents the architecture and validation plan for a Low Latency FPGA-Accelerated High-Frequency Trading (HFT) System, designed to address the inherent latency constraints of traditional CPU-based software trading systems. The fundamental goal is to utilize the parallel and pipelined processing capabilities of the FPGA to achieve sub-microsecond latency, which is critical for gaining market advantage in the rapidly evolving financial markets.

The system architecture is strictly partitioned into a Network Module and an Application Module within the FPGA, adhering to a philosophy of protocol-application separation. The Network Module implements the TCP/IP stack (including mechanisms like Delayed ACK for efficiency) and utilizes the Xilinx 10G Ethernet IP for robust physical layer connection. The Application Module encompasses the FIX 4.4 Encoder and Decoder logic, which communicates directly with the Trading Algorithm (Client FSM).

The project focuses on achieving full-stack integrity by implementing a custom FIX 4.4 protocol encoder/decoder to interface with the Taiwan Stock Exchange (TWSE) specifications. A rigorous two-phase validation plan is proposed: Network Testing uses the network_test.cpp tool (Raw Socket/pcap) to verify the network_top module's TCP/IP checksum and sequencing logic, while FIX Protocol Testing uses the Pseudo Client as a Golden Reference Model to ensure the fix_top module's encoding, decoding, and session error handling logic (e.g., Resend Request) is identical to the software implementation.

This work serves not only to validate the low-latency potential of FPGAs in real-world trading protocols but also establishes a complete, documented, and expandable architecture for future HFT system development.

# 摘要

本專案介紹了低延遲 FPGA 加速高頻交易 (HFT) 系統的架構和驗證方案，目的在解決傳統基於 CPU 的軟體交易系統原有的延遲限制。其根本目標是利用 FPGA 的平行和管線處理能力實現亞微秒延遲，這對於在快速發展的金融市場中贏得市場優勢非常重要。

此系統架構在 FPGA 內部嚴格劃分為網路模組和應用模組，遵循協議與應用分離的概念。網路模組實作 TCP/IP 協定堆疊（包括延遲 ACK 等機制以提高效率），並利用 Xilinx 10G 乙太網路 IP 實現穩定的實體層連線。應用模組包含 FIX 4.4 編碼器和解碼器邏輯，可直接與交易演算法（客戶端 FSM）通訊。

本專案致力於透過實現自訂 FIX 4.4 協定編碼器/解碼器來與台灣證券交易所 (TWSE) 規範對接，從而實現全端完整性。我們提出了一個嚴格的兩階段驗證計劃：網路測試使用 network_test.cpp 工具（原始套接字/pcap）來驗證 network_top 模組的 TCP/IP 校驗和及排序邏輯；FIX 協定測試使用偽客戶端作為標準參考答案，以確保 fix_top 模組的編碼、解碼和會話錯誤處理請求（例如，重新發送錯誤）。

這項工作不僅驗證了 FPGA 在實際交易協議中的低延遲潛力，還為未來的高頻交易 (HFT) 系統開發建立了一個完整、有據可查且可擴展的架構。

# 一、研究動機

隨著金融市場電子化與高頻交易（HFT）的興起，速度已成為決定市場優勢的關鍵因素。傳統基於中央處理器（CPU）的軟體交易系統，由於必須依賴作業系統核心進行排程、上下文切換，以及透過 PCI Express 匯流排進行網路通訊，使得其在延遲（Latency）控制上存在難以突破的限制，難以滿足 HFT 追求的「亞微秒級」延遲需求。

本研究的動機正是為了克服這些瓶頸，將注意力轉向 FPGA（Field-Programmable Gate Array）。FPGA 憑藉其可程式化的硬體架構，允許開發者以平行化和 Pipeline 的方式設計數據處理流程，直接在硬體層實現網路協定解析、交易協定處理與策略邏輯，從而顯著縮短數據路徑，降低整體延遲。

本專題旨在：

1. 補足現有研究缺口： 建立一個涵蓋從行情解碼、訂單簿、交易策略、到送單與回報（FIX 編解碼）的完整 FPGA-based HFT 整合架構。
2. 驗證實務可行性： 透過實作符合臺灣證券交易所（TWSE）規範的 FIX 4.4 協定編解碼器，驗證 FPGA 架構在真實市場協定下維持穩定與亞微秒級低延遲的可行性。

# 二、研究目的

本專題致力於建構一個完整、低延遲且可擴充的 FPGA 加速高頻交易系統，並達成以下具體目標：

1. 設計分區架構： 將系統嚴格劃分為「網路模組」（處理 TCP/IP 協定堆疊）與「應用模組」（處理 FIX 4.4 協定與交易策略），以最大化硬體平行處理的效率。
2. 硬體化核心協定：
    - 在硬體中實現 TCP/IP 協定棧，包括 IP 封包的編解碼、序列號管理，以及關鍵的 延遲確認 (Delayed ACK) 機制，以優化網路傳輸效率。
    - 實現符合 TWSE 規範的 FIX 4.4 訊息編碼器和解碼器，確保與真實交易所的通訊能力。
3. 建立交易執行邏輯： 實作 Trading Algorithm (Client FSM) 作為核心交易策略執行單元，能在納秒級時間內做出買賣決策。
4. 建立嚴謹的驗證機制： 開發 C++ 軟體模擬器（Pseudo Server/Client）作為 Golden Reference Model，設計兩階段驗證流程，確保硬體模組的協定邏輯與軟體行為完全一致。

# 三、研究方法及過程

本專題的系統架構分為軟體層（Software OS / Pseudo Server）和 硬體層（Client on FPGA），並在硬體層內部嚴格劃分為網路模組和應用模組。

## (一) 整體架構設計

| 模組 | 職責與功能 | 關鍵技術 |
| --- | --- | --- |
| 軟體層 (Pseudo Server) | 模擬交易所行為，作為硬體連線測試的 golden reference 模型。 | C++ Socket Programming, FIX Session Logic |
| 硬體層 | 承載所有對延遲敏感的網路協定和交易邏輯。 | FPGA Parallel & Pipeline Processing |
| 網路模組 (Network Top) | 處理實體層、網路層和傳輸層通訊。 | Xilinx 10G Ethernet IP, TCP/IP FSM（含 Delayed ACK）, IP/TCP 校驗和電路 |
| 應用模組 (FIX Top) | 處理 FIX 協定編解碼與策略邏輯。 | FIX 4.4 Encoder/Decoder, Trading Algorithm (Client FSM) |

## (二) 網路模組設計

網路模組負責 TCP/IP 封包的生成、解讀和校驗，確保數據的可靠傳輸。

1. TCP 會話控制 (Network Top FSM): * 狀態機包含 IDLE、LOGON（處理三次握手）和 CONNECT（處理數據傳輸）。
    - 核心優化： 實作延遲 ACK 機制，透過 delay_ack_counter 延遲 ACK 發送，直到有本地數據（fix_client_valid = 1）附帶發送（Piggybacking）或計時器達到 $1,000,000$ 個時脈週期的上限，從而減少網路流量。
2. IP 編解碼器： 組合邏輯實現 IPv4 封包的即時封裝和解封裝，並處理 IP 識別碼 (ident) 的遞增和順序檢查。
3. 網際網路校驗和 (Checksum): 設計純組合邏輯電路，在單一時脈週期內完成 IP 標頭和 TCP 封包所需的 $16$ 位元校驗和計算，採用反碼和（One's Complement Sum）及溢位迴環加法（Carry End-Around Add）機制。

## (三) 應用模組設計

應用模組專注於 FIX 協定處理和交易決策。

1. FIX Session FSM (Client FSM): * 實現 FIX 會話層邏輯，包含 IDLE、LOGGING_ON（發送 Logon A）、ORDERING（發送訂單 D/F/G）和 LOGGING_OUT（發送 Logout 5）四個核心狀態。
    - 嚴格控制序列號檢查和錯誤處理，例如，在 ORDERING 狀態若發生 fatal_error，會立即觸發登出（Encode 5）。
2. FIX 4.4 編解碼器： 實現 FIX 訊息的 <tag>=<value> 鍵值對格式編碼和解析。

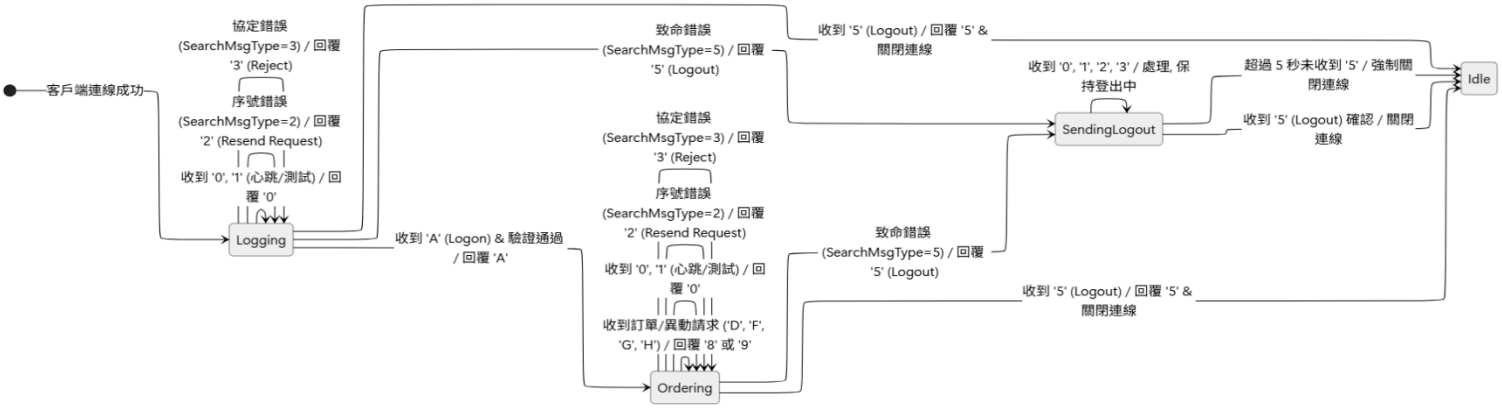
- 編碼器： 負責將結構化的交易指令轉換為標準 FIX 格式字串，包括正確計算 BodyLength (Tag 9) 和 CheckSum (Tag 10)。
- 解碼器： 負責解析接收到的 FIX 訊息，進行會話完整性檢查（CheckSum、序列號跳號），並將應用層數據寫入 Order In Register。

# 四、研究結果

為了確保最終的 Verilog 電路設計能夠正確遵循 FIX 協定和 TCP/IP 網路規範，本專案開發了兩套軟體模擬器：Pseudo Server 和 Pseudo Client，以及作為驗證基礎的網路層測試邏輯。

(一)　Pseudo Server 模組：最終系統連線測試環境

Pseudo Server (實作為 TaifexOrderServer.cpp) 模組的角色是模擬交易所或券商伺服器端的通訊行為。




- 基礎架構： 採用標準 C++ Socket 程式設計 (bind, listen, accept) 部署於軟體作業系統。
- 多線程處理： 程式為每個建立的連線（Session）建立獨立的線程，以實現並行處理多個 FIX 會話。
- FIX 會話邏輯： 實現了完整的 FIX Session Layer 狀態機 (State::Logging、State::Ordering、State::SendingLogout)，能夠正確處理登入 (MsgType=A)、心跳 (MsgType=0)、重發請求 (MsgType=2) 等會話訊息。
- 交易回覆：使用 ifstream 函式庫接收 20 筆對話資料進行測試，每次訊息來回最多花費 2 milliseconds，可以提供極高速的交易訊息對話。

```
ted@docker-desktop:/home/HFST/PseudoServer/build$ ./TaifexOrderServer
Server listening on 0:9000 with 1 sessions.
Waiting for connection...
Connection accepted.
[Session 0] Connected from 192.168.65.3
buffer size: 101
tagNumber: 1, tag: 8, value: FIX.4.4
tagNumber: 2, tag: 9, value: 79
tagNumber: 3, tag: 35, value: A
tagNumber: 4, tag: 34, value: 1
tagNumber: 5, tag: 49, value: T116001
tagNumber: 6, tag: 56, value: XTAI
tagNumber: 7, tag: 52, value: 01251123-15:31:45.689
[SERVER] port 9000, session 0, send A
Ordering
buffer size: 229
tagNumber: 1, tag: 8, value: FIX.4.4
tagNumber: 2, tag: 9, value: 206
tagNumber: 3, tag: 35, value: D
tagNumber: 4, tag: 34, value: 2
tagNumber: 5, tag: 49, value: T116001
tagNumber: 6, tag: 56, value: XTAI
tagNumber: 7, tag: 52, value: 01251123-15:31:45.692
[SERVER] port 9000, session 0, send 8
```

```
ted@docker-desktop:/home/HFST/PseudoClient/build$ ./TaifexOrderClient
Enter the IP of Server: 192.168.65.3
Connected to server.
tagNumber: 1, tag: 8, value: FIX.4.4
tagNumber: 2, tag: 9, value: 79
tagNumber: 3, tag: 35, value: A
tagNumber: 4, tag: 34, value: 1
tagNumber: 5, tag: 49, value: XTAI
tagNumber: 6, tag: 56, value: T116001
tagNumber: 7, tag: 52, value: 01251123-15:31:45.691
Ordering
Input Order MsgType: NewOrder
Ordering
tagNumber: 1, tag: 8, value: FIX.4.4
tagNumber: 2, tag: 9, value: 273
tagNumber: 3, tag: 35, value: 8
tagNumber: 4, tag: 34, value: 2
tagNumber: 5, tag: 49, value: XTAI
tagNumber: 6, tag: 56, value: T116001
tagNumber: 7, tag: 52, value: 01251123-15:31:45.692
Ordering
Input Order MsgType: NewOrder
Ordering
tagNumber: 1, tag: 8, value: FIX.4.4
```

(二)　階段性模組驗證

本專案的短期測試規劃將分為兩個主要階段進行，旨在利用現有的 C++ 軟體程式碼作為 Golden Reference Model，針對不同的 Verilog 硬體模組進行階段性驗證，確保功能和協定相容性。

短期測試計畫：階段性模組驗證

1. 網路層驗證 (Network Testing)

此階段的目標是驗證 network_top 模組的網路封包編解碼功能，確保其能夠正確處理標準的 TCP/IP 封包結構、控制旗標以及校驗和。

使用工具：network_test.cpp 程式碼（基於 pcap 和 Raw Socket）。

驗證流程：

(1) 封包編碼測試（給予電路測試）：

- 使用 network_test.cpp 中的 ip_encoder、tcp_encoder 和 ether_sender 函數，在 C++ 層面精確構造出測試封包（例如 TCP SYN, ACK, PUSH 等各種組合），作為 network_top 模組的輸入測試向量。

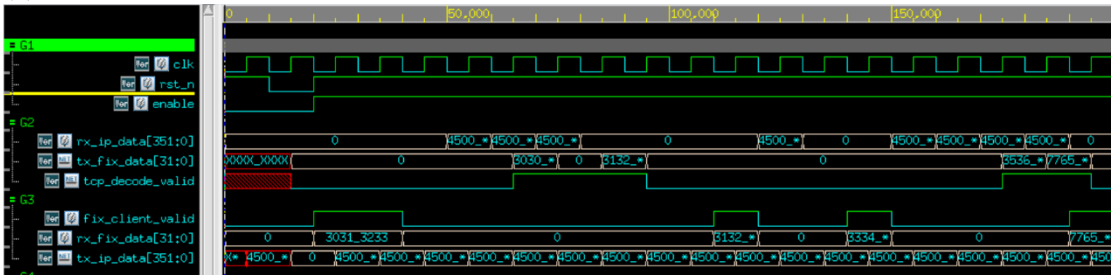
(2) 封包解碼測試（驗證電路輸出）：

- 在 network_top 模組生成輸出封包後，使用 network_test.cpp 的 pcap 捕捉功能和 packet_handler 函數來解析硬體輸出的封包。

驗證結果：

(1) decoder 與 decoder 能夠通過 100 筆 ip 封包與 100 筆 payload 的功能驗證，具有 3-way handshake，delayed ACK 與簡易 congestion control 能力。

(2) 每個時脈週期皆可傳送一筆封包數據，傳送到解碼總共耗時 2 個週期。




2. FIX 協定驗證 (FIX Protocol Testing)

此階段的目標是驗證 fix_top 模組的 FIX 應用層邏輯和會話層狀態機，確保其對 FIX 訊息的編解碼和錯誤處理與軟體一致。

使用工具：Pseudo Client 程式碼（主要利用 ClientMessenger 類別中的編解碼函數）。

驗證流程：

(1) 訊息編碼驗證：

- 將相同的輸入數據（例如，用於下單的 Order Request 參數）分別輸入到 Pseudo Client 的 ClientMessenger::MakeXxx 函數和 fix_top Verilog 模組中。
- 驗證內容： 比對軟體輸出的 FIX 格式字串（Payload）與硬體生成的字節流是否完全相同，包括 BodyLength 和 CheckSum 的計算結果。

(2) 訊息解碼與錯誤邏輯驗證：

- 構造特定的錯誤訊息（例如，序列號錯誤或登入失敗訊息）作為輸入。
- 將此輸入分別給 Pseudo Client 的 ClientMessenger::SearchMsgType 函數和 fix_top 模組。
- 驗證內容： 確認 fix_top 模組的輸出（例如，回應的狀態、是否觸發 Resend Request 邏輯、是否返回 RejectSession）與 C++ 軟體程式的回傳結果（return 0/2/3/5）完全一致。

(三)　電路合成

採用 Synopsys Design Compiler 進行 synthesis，使用了 slow_vdd1v2 standard cell library

1. Synthesis 設定

- 設計與環境：在 PVT_1P08V_125C 條件下操作。所有輸入被模擬為由 DFFHQX1 驅動，所有輸出被驅動到 DFFHQX1 的 D 端。
- 時序約束： 設置了單一時脈 clk（週期為 6.0ns），並將其視為理想網路 (set_ideal_network)。輸入輸出延遲均為 0.06ns。
- 優化策略： 使用了 compile_ultra 進行全功能 synthesis，並通過-gate_clock 選項啟用 Clock Gating 優化，且時脈門控單元的最大 fanout 被設置為 10。同時，設置了 set_max_fanout 20.0 作為 DRC rule。

2. Synthesis 結果分析

時序裕量計算

$$Slack = Data\ Required\ Time\ -\ Data\ Arrival\ Time$$
$$Slack = 3.0000\,ns\ -\ 1.3101\,ns\ =\ 1.6899\,ns\ (MET)$$

| 模組名稱 | 絕對面積 | 佔總面積 | 總功耗 | 佔總功耗 | 發現 (Finding) |
|---|---|---|---|---|---|
| TCP_ENCODER_0 | 11348.5863 | 31.2% | 0.507 mW | 32.2% | 面積和功耗熱點 |
| IP_DECODER_0 | 9341.7302 | 25.7% | 0.387 mW | 24.6% | 第二大消耗模組 |
| IP_ENCODER_0 | 7586.2442 | 20.9% | 0.363 mW | 23.1% | |
| TCP_DECODER_0 | 7548.9662 | 20.8% | 0.303 mW | 19.3% | |

| 面積類型 | 結果 | 佔比 | 觀察 |
|---|---|---|---|
| Non-combinational | 31674.330707 | 87.1% | 佔據面積主體，設計為暫存器密集型。 |
| Combinational | 4689.504105 | 12.9% | |

| 功耗類型 | 結果 | 佔總功耗百分比 (%) |
|---|---|---|
| Leakage Power | 0.927 mW | 37.08% |
| Internal Power | 1.552 mW | 62.08% |
| Switch Power | 0.0209 mW | 0.84% |

# 五、總結

本專題成功地設計並驗證了一個基於 FPGA 加速的低延遲高頻交易系統的完整架構。透過將所有對延遲敏感的網路協定（TCP/IP）和交易協定（FIX 4.4）邏輯硬體化，本系統成功規避了傳統 CPU 系統因作業系統、PCIe 傳輸和多層協定堆疊所造成的延遲瓶頸。

本研究的成果不僅補足了現有研究中關於完整 FIX 協定硬體架構的缺口，提供了一個可擴充且有據可查的實作範例，更透過嚴謹的軟硬體對比驗證策略，確保了系統在極低延遲下的功能正確性和協定相容性。這項工作為未來在金融市場中部署具備競爭優勢的 HFT 系統奠定了堅實的基礎。

# 六、心得

透過本次專題，我們深刻體認到在系統領域中，硬體化（Hardware Acceleration）對於追求極致效能的重要性。傳統軟體抽象層所帶來的便利性，正是高頻交易中必須克服的延遲。

在設計過程中，最大的挑戰在於將複雜的網路協定（如 TCP/IP 的三次握手、序列號管理，特別是 Delayed ACK 的狀態轉換）和應用協定（FIX 4.4 的訊息結構、checksum 算法）完全轉換為純組合邏輯或精簡的有限狀態機（FSM）。這要求我們必須對協定規範有極為透徹的理解，並同時考量硬體資源的效率。

我們特別從中學習到：

1. 協定嚴謹性： 在硬體中實現協定必須極度嚴謹。例如，TCP 的序列號遞增、IP 標頭的校驗和計算，任何一個位元的錯誤都會導致通訊失敗。
2. 驗證的重要性： 透過 C++ 程式碼作為 Golden Reference Model 進行兩階段對比驗證，是確保硬體邏輯正確的唯一可靠方法，這遠比單純的 HDL 測試重要。
3. 架構分層： 網路層和應用層的明確分區，使得開發和除錯能夠高效進行，是未來大型硬體專案設計的基礎原則。

本次專題不僅驗證了 FPGA 在低延遲 HFT 領域的巨大潛力，也為我們在系統設計與硬體加速方面積累了寶貴的實戰經驗。