Smart Network Switch

Group 9

Mahmoud Kharsa
Mark Bugaisen
David Nguyen

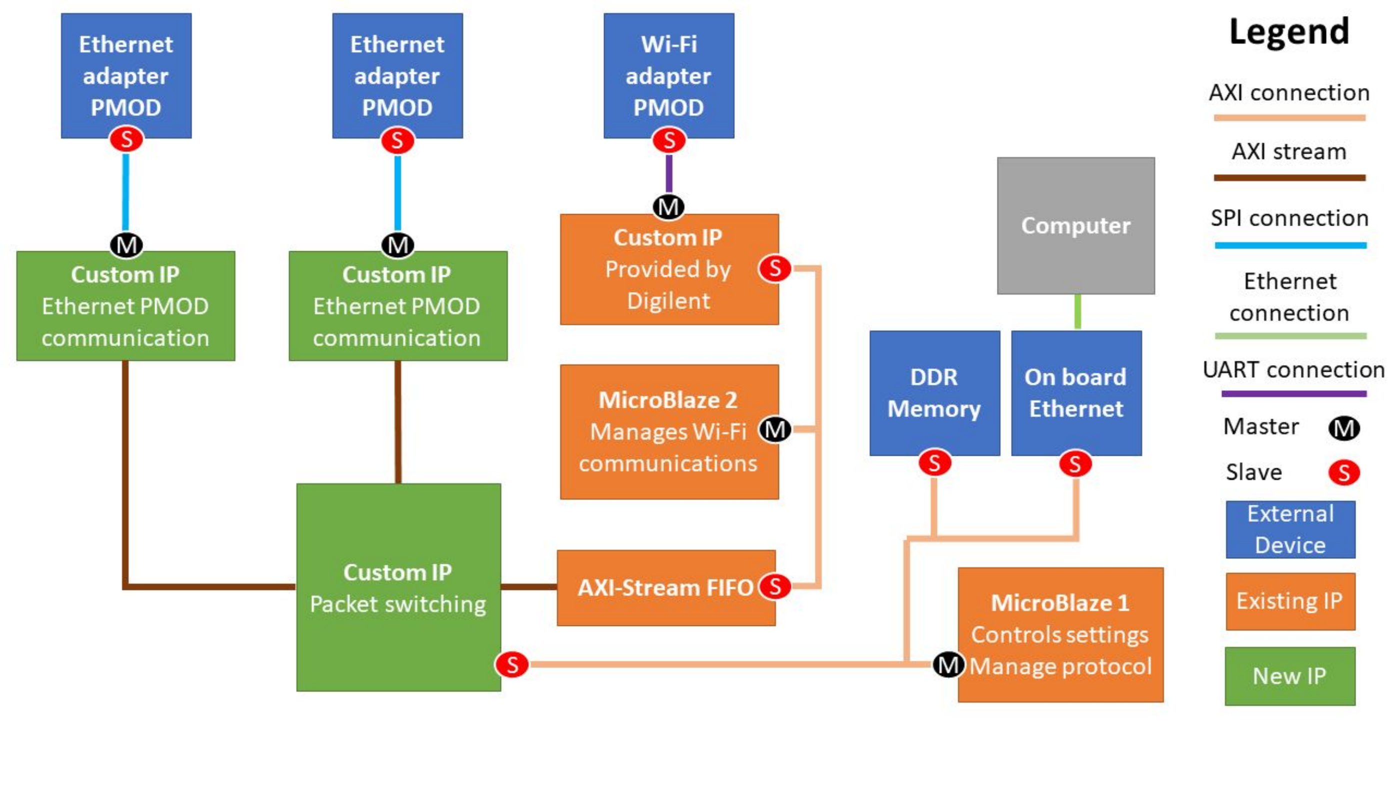
Project Overview

- Forward network traffic to the right target
- Done by building an address-port table

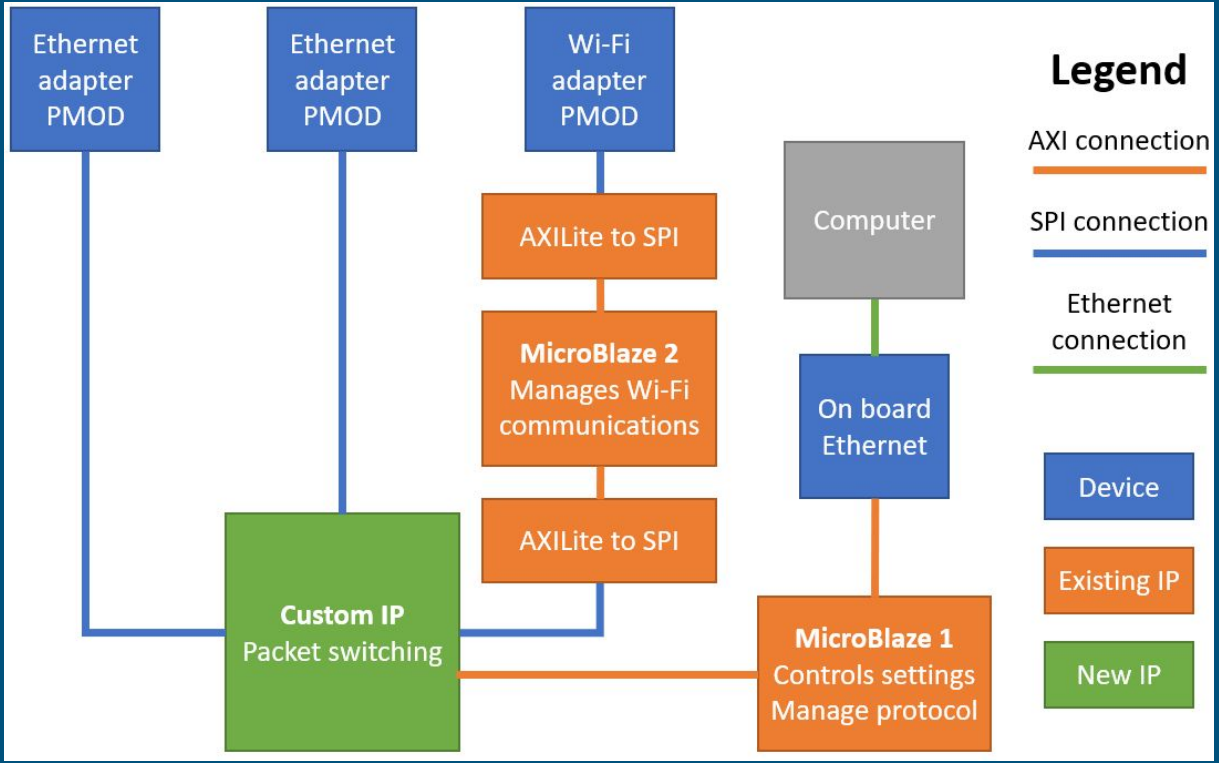
Requirements

● FPGA routes packets to the correct destination device (in hardware)	✓
● Support two devices connected across Ethernet	✓
● Support one device connected across Wi-Fi	~
● Support IPv4	✓
● Remotely configurable	✓
● Has monitoring capabilities from a PC connected across Ethernet	✓
● Minimum Throughput: 1 Mbps 10 Mbps per port	✓
● Maximum Latency: 100 ms 50 ms max bound case	✓

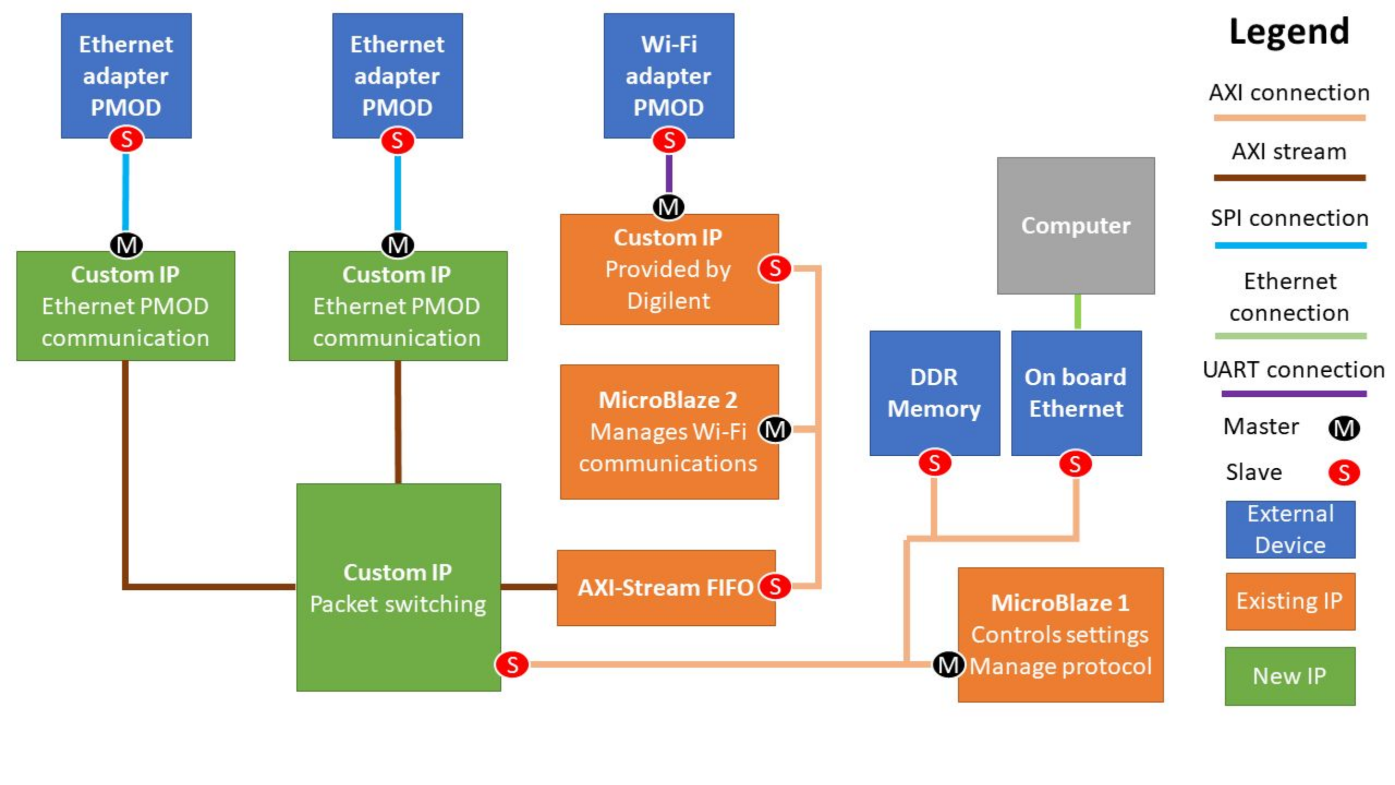
Block Diagram of Final System



Block Diagram from Proposal

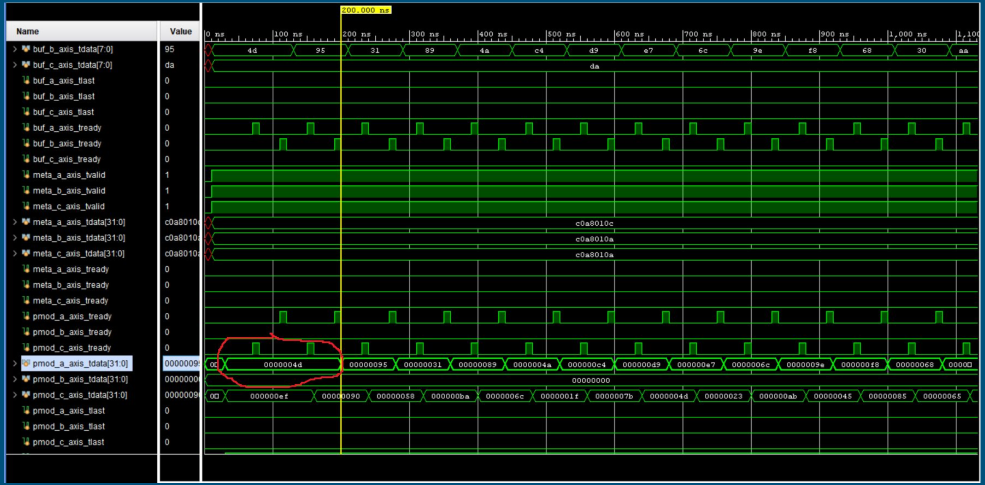


Block Diagram of Final System



Challenges Encountered

- The Wi-Fi only support UDP/TCP level communication
  - Only delivers packet content, no headers.
- Wi-Fi microblaze DDR memory
- Endianness
- Packet switching tvalid, tdata, tlast signals



Custom IPs created (Hardware)

- Packet Switcher IP (Custom IP)
  - Designed entirely from scratch, utilizes AXI Stream Data FIFOs
  - Interfaces with AXI Stream (two for each port) and AXI Lite (remote config)
  - Extracts metadata (ARP, IPv4, IP address)
  - Drops unknown packets
  - Directs traffic across three ports in round robin

- NIC100 PMOD AXI Stream IP (Custom IP)
  - Based off a git implementation (https://github.com/carljohnsen/Pmod-NIC100)
  - Modified to interface with AXI Stream
  - Added additional commands for initializing PMOD in promiscuous mode

Source Code created (Software)

- Wi-Fi PMOD Microblaze 2 code (ESP32)
  - Based off a git implementation https://github.com/Digilent/vivado-library/tree/master/ip/Pmods/PmodESP32_v1_0/
  - Modified to relay packets between the PMOD and the AXI Stream interface to the Packet Switcher IP
- LWIP Server for Remote configuration over onboard Ethernet
  - Modified echo.c to accept commands (iptable -l, iptable -s, status)
- Client Python script for issuing commands to LWIP Server
- Python scripts for removing/adding UDP wrappers (Wi-Fi connected device)

Overall Design Process

- Overall project was split into different modules and divided amongst us
  - At the start, David did the Wi-Fi, Mark did the Ethernet, Mahmoud did the custom IP
  - At the end, David did the Wi-Fi, Mark did integration, Mahmoud did the remote config.
  - Ensured that at any point in time, each person was doing something
- Thorough Unit-Testing of modules
  - Simulation used to validate the packet switching IP
  - ILAs/debug print statements from the Microblaze used to validate the PMODs
  - Helped catch many bugs early on in the project before the final integration

What We Learned…

- How a basic router works and how it is implemented in hardware
    - For the people in the team that haven’t taken Computer Networks…
        - How UDP communication works
        - How to use Wireshark to view incoming packets
- Not to underestimate integration....
    - Initially proposed 1-2 spare weeks for integration, ended up using ALL of that time
    - Stresses the importance of unit-testing to catch bugs early on
    - ILAs were very helpful in isolating and root-causing bugs

NOW FOR THE FINAL DEMO!