The *Nand2Tetris* project is a well-known course that teaches students how to build a complete computer system from the ground up, starting with a single NAND gate. This project attempts to translate that theoretical knowledge into practice by implementing key components of the *Nand2Tetris* hardware architecture using an FPGA platform.

Objectives

- To implement all hardware modules defined in Part I of the *Nand2Tetris* course using Verilog, following a bottom-up design approach.
- To develop custom testbenches for each module to ensure correctness, modularity, and reusability.
- To gain hands-on experience in writing synthesizable HDL code and understanding its real-world implications on FPGA hardware.
- To prepare the groundwork for running interactive applications, such as a simplified Tetris game, on the FPGA as a final demonstration of system capability.

Tools and Technologies

- **FPGA Platform:** Basys 3 Artix-7 FPGA Trainer Board
- **HDL Used:** Verilog
- **Design and Simulation Tool:** Vivado 2024.2 (including XSIM for simulation)
- **Reference Material:** *The Elements of Computing Systems* by Nisan and Schocken

Project Scope and Methodology

This project closely follows the structure of *Nand2Tetris* Part I, implementing each hardware module in Verilog and testing it using custom testbenches. The work is organized as follows:

- **Project 1: Boolean Logic**
  Implemented fundamental gates (NAND, NOT, AND, OR, XOR, etc.) using only the NAND gate as the primitive building block. Verified each gate using Verilog testbenches.
- **Project 2: Combinational Chips**
  Designed multiplexers, demultiplexers, adders, and other combinational circuits. Focused on hierarchical design and modular reuse of previously implemented gates.
- **Project 3: Sequential Logic**
  Built flip-flops, registers, counters, and memory units (RAM8, RAM64, etc.) using clocked logic. Verified proper state transitions and data storage over clock cycles.

- **Project 4: The ALU**
  Developed a 16-bit Arithmetic Logic Unit capable of handling arithmetic and logical operations based on control inputs. Emphasized correct implementation of zeroing, negation, and comparison outputs.
- **Project 5: The Computer Architecture (CPU)**
  Currently in progress: constructing the Hack CPU by integrating the ALU, instruction decoding logic, control unit, and program counter. Includes instruction execution and memory interface.
- **Project 6: Machine Language Programs**
  Once the CPU is complete, programs written in Hack Assembly will be used to test real execution flow. This phase sets the foundation for running a Tetris game or similar application on FPGA hardware.

Each module is first designed and tested in simulation.

Challenges Encountered

- **Steep Learning Curve in Verilog:**
  As a first-time Verilog user, adapting to hardware-oriented thinking and writing synthesizable code required a shift in mindset. Early modules involved frequent trial and error as I familiarized myself with Verilog syntax, simulation practices, and testbench design.
- **Navigating the Web IDE's CPU Emulator:**
  While the *Nand2Tetris* Web IDE is intuitive for basic modules, getting familiar with the built-in CPU Emulator posed some initial difficulties. Understanding how to load programs, trace execution, and verify control logic through the emulator required some hands-on experimentation.

Acknowledgments

- MARVEL UVCE
- Noam Nisan and Shimon Schocken, creators of the Nand2Tetris course.