FPGA-Based Digital Compass Using the Nexys A7-100T and Pmod CMPS2 Sensor

Christian Vanegas
College of Engineering
California State University, Long Beach
Long Beach, USA
Christian.Vanegas02@student.csulb.edu

Nathan Sarkozy
College of Engineering
California State University, Long Beach
Long Beach, USA
Nathan.Sarkozy@student.csulb.edu

Kaiya Hayashida
College of Engineering
California State University, Long Beach
Long Beach, USA
Kaiya.Hayashida@student.csulb.edu

Abstract – The purpose of this project is to design and implement a real-time digital compass on the Nexys A7-100T FPGA board. We will determine and display the precise heading angle in degrees relative to magnetic north using the Pmod CMPS2 magnetometer. We will utilize the I²C communication protocol to process the raw X, Y, and Z-axis magnetic data and compute the correct directional heading. The result will be the angle in degrees delivered to the board's seven segment display for real-time feedback. This project demonstrates th e u se of digital logic design principles to create a hardware-based embedded system for sensor integration and real-time computation.

I. INTRODUCTION

This digital compass project applies the principles of combinational and sequential circuit design to create a fully functional hardware-based navigation system. The FPGA will acquire magnetic field data from the sensor, process it through custom logic to compute the real-time heading angle relative to magnetic north, and display the result on the board's seven-segment display. By integrating sensor interfacing, data processing, and real-time visualization, this project demonstrates how foundational digital logic concepts can be combined to implement a practical and sophisticated embedded system.

II. BACKGROUND & PRELIMINARIES

The concept of a digital compass relies on measuring the Earth's magnetic field to determine orientation relative to magnetic north. Modern electronic compasses typically rely on magnetometer sensors that output analog or digital readings corresponding to the magnetic field's strength along the X, Y, and Z axes. By processing these readings, the heading angle can be calculated using trigonometric relationships. The Pmod CMPS2 magnetometer is a 3-axis magnetometer based on the LSM303DLHC chip, which communicates with external devices via the I²C serial interface. This allows for precise and easily accessible magnetic field data.

In this project, the Nexys A7-100T FPGA serves as the central processing platform for real-time computation and display. Unlike microcontrollers, FPGA's allow for parallel digital logic design, making them ideal for custom hardware modules that handle sensor communication, signal processing, and display control simultaneously. Implementing the I²C protocol in Verilog will enable direct communication between the FPGA board and the Pmod CMPS2 sensor, while the computed angle will be displayed using the board's seven-segment display interface. An additional SPI driver will interface with the accelerometer for tilt measurement and compensation.

To achieve accurate directional readings regardless of device orientation, a tilt compensation circuit will process data from both the magnetometer and accelerometer to correct for roll and pitch angles. The heading calculation circuit will then use the compensated magnetic field data to compute the true heading relative to magnetic north. Finally, the seven-segment display driver (which will require a time-multiplexing scheme due to shared segment lines) will present the calculated heading angle in real time. Together, these modules form a complete FPGA-based embedded system capable of sensor integration, tilt correction, real-time angle computation, and visual output.

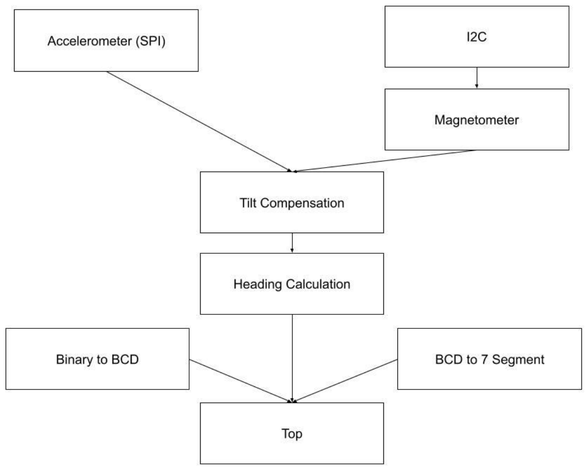


Figure 1. Design Diagram

III. IMPLEMENTATION

The binary_to_bcd module converts a 10-bit binary input (ranging from 0 to 511) into its three-digit Binary-Coded Decimal (BCD) representation. Inside the combinational always block, it checks whether the input exceeds 511; if so, it outputs the digits “5-1-1” as a clamped maximum value. Otherwise, it calculates the hundreds, tens, and ones digits using division and modulus operations: the hundreds digit is binary / 100, the tens digit is the tens place of the remainder, and the ones digit is the final remainder. The resulting BCD digits can then be used directly for display on the seven-segment display.

The bcd_to_7seg module converts a 4-bit BCD (Binary-Coded Decimal) input into the corresponding 7-segment display pattern for digits 0 through 9. Inside the combinational always block, it uses a case statement to map each BCD value to its predefined 7-bit segment configuration for a common-cathode display, where each bit controls one of the seven LED segments. If the input is outside the valid 0–9 range, the module outputs 1111111, turning all segments off.

The i2c_master module functions as the custom I²C master used to interface the Nexys A7-100T FPGA with the Pmod CMPS2 magnetometer, enabling reliable communication for retrieving the sensor’s magnetic field measurements. It implements the complete I²C protocol in hardware, including generation of START, RESTART, and STOP conditions, transmission of device and register addresses, and support for multi-byte read or write operations. The controller uses a finely timed state machine and open-drain signaling to match the electrical behavior of the I²C bus, while status signals such as busy, done, rd_valid, and error allow the rest of the system to coordinate data transfers. Within the compass project, this module serves as the foundation for acquiring raw X, Y, and Z magnetometer data in real time, ensuring that heading calculations are performed on accurate and consistently sampled sensor values.

The i2c_master testbench provides a controlled simulation environment for verifying the functionality, timing behavior, and fault handling of the custom I²C master. It generates a stable 100 MHz clock, applies reset and start conditions, and drives all necessary control signals to exercise both read and write transactions. A fully modeled behavioral I²C slave is included to emulate the Pmod CMPS2 device, supporting address recognition, register addressing, ACK/NACK responses, multi-byte data transfers, and error cases such as incorrect addressing or slave inactivity. The testbench captures all bytes returned by the master, logs communication events, and evaluates results using structured test tasks that automatically record pass/fail outcomes. By simulating a range of scenarios, including single-byte writes, multi-byte reads, alternating data patterns, varying slave addresses, and intentional protocol errors, the module ensures that the I²C master behaves reliably across all expected operating conditions. This verification framework is essential for validating real-time sensor communication before deploying the design to the Nexys A7 FPGA.

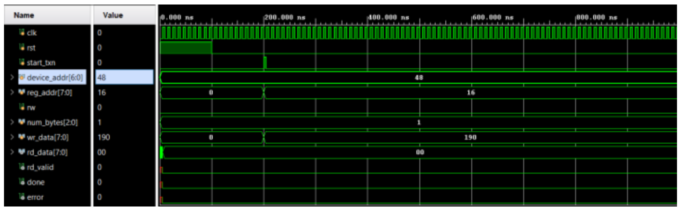
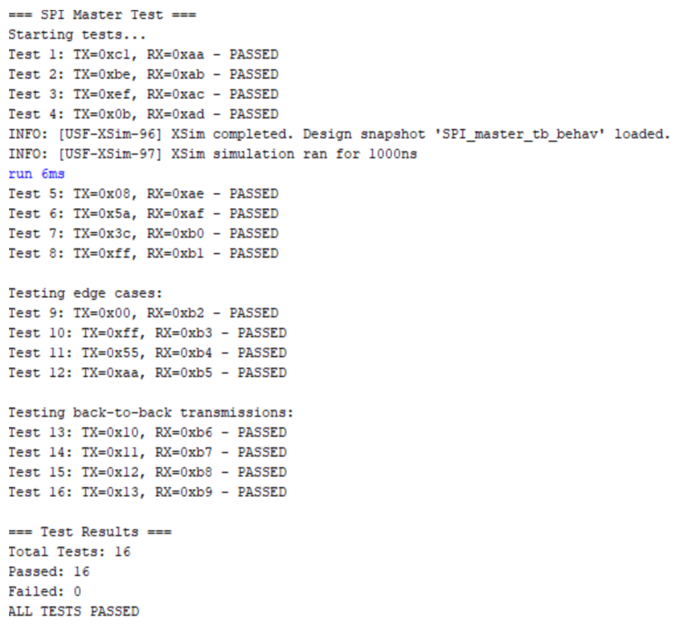


Figure 2. I2C Master Testbench Simulation Waveform

The SPI_master module implements the SPI master interface responsible for configuring and retrieving accelerometer data from the SPI-based motion sensor used alongside the FPGA in the digital compass system. It generates a 1 MHz SPI clock from the 4 MHz system clock, manages chip-select timing, and sequences all SPI transactions through a structured finite-state machine. The module first issues a multi-byte write sequence to configure the sensor’s operating mode, then periodically performs read transactions to collect six bytes containing the X, Y, and Z axis measurements. Incoming data is sampled on MISO, stored into axis-specific registers, and later compressed into a 15-bit output containing the sign bit and upper data bits of each axis for downstream processing.

The SPI_master testbench verifies the functionality of the SPI_master module by simulating realistic SPI communication between the FPGA and an accelerometer sensor. It generates the 4 MHz system clock, drives controlled MISO responses based on the internal state of the SPI master, and feeds predefined 16-bit X, Y, and Z accelerometer samples that mimic what the real sensor would output over SPI. The testbench uses a structured task, run_orientation_test, to automate multiple orientation scenarios by extracting the expected 5-bit values from each axis, applying the simulated sensor data, waiting for the SPI transaction to complete, and comparing the module's 15-bit output against the expected combined value. It logs detailed results for each test, including bit-level verification of the X, Y, and Z fields, and tracks total passes and failures, simulating a self-checking testbench.



```
=== SPI Master Test ===
Starting tests...
Test 1: TX=0xc1, RX=0xaa - PASSED
Test 2: TX=0xbe, RX=0xab - PASSED
Test 3: TX=0xef, RX=0xac - PASSED
Test 4: TX=0x0b, RX=0xad - PASSED
INFO: [USF-XSim-96] XSim completed. Design snapshot 'SPI_master_tb_behav' loaded.
INFO: [USF-XSim-97] XSim simulation ran for 1000ns
run 6ms
Test 5: TX=0x08, RX=0xae - PASSED
Test 6: TX=0x5a, RX=0xaf - PASSED
Test 7: TX=0x3c, RX=0xb0 - PASSED
Test 8: TX=0xff, RX=0xb1 - PASSED

Testing edge cases:
Test 9: TX=0x00, RX=0xb2 - PASSED
Test 10: TX=0xff, RX=0xb3 - PASSED
Test 11: TX=0x55, RX=0xb4 - PASSED
Test 12: TX=0xaa, RX=0xb5 - PASSED

Testing back-to-back transmissions:
Test 13: TX=0x10, RX=0xb6 - PASSED
Test 14: TX=0x11, RX=0xb7 - PASSED
Test 15: TX=0x12, RX=0xb8 - PASSED
Test 16: TX=0x13, RX=0xb9 - PASSED

=== Test Results ===
Total Tests: 16
Passed: 16
Failed: 0
ALL TESTS PASSED
```

Figure 3. SPI Master Testbench Console Output

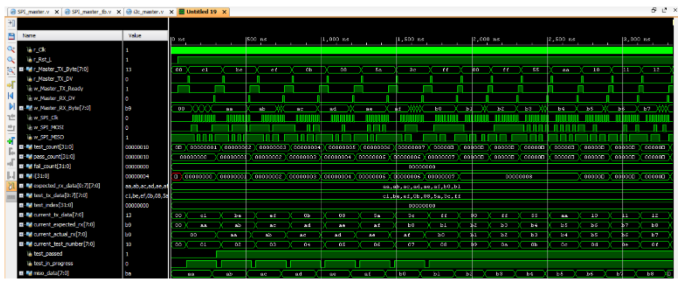


Figure 4. SPI Master Testbench Simulation Waveform

The magnetometer_driver module serves as the high-level controller that manages all communication and data processing required to obtain magnetic field measurements from the Pmod CMPS2 magnetometer over I²C. It coordinates the full measurement sequence by issuing the appropriate control-register commands to trigger a reading, waiting the required measurement time, and then performing a six-byte read to retrieve the X, Y, and Z axis outputs. The module assembles the incoming bytes into 16-bit values, converts the magnetometer's unsigned output format into signed measurements by subtracting the sensor's midpoint offset, and asserts a one-cycle data_valid signal when fresh data is ready for use elsewhere in the system. Internally, a finite-state machine manages timing, byte collection, error detection, and interface handshaking with the underlying I²C master, while additional debug signals expose both the I²C and driver states for testing and verification.

The mag_driver testbench verifies the full operation of the magnetometer_driver module by simulating realistic I²C communication with a virtual Pmod CMPS2 sensor and evaluating the driver's ability to correctly retrieve and process magnetic field readings. It generates the 100 MHz system clock, handles reset sequencing, and instantiates a detailed behavioral I²C slave model that emulates the magnetometer's addressing, register protocol, ACK/NACK behavior, and multi-byte data transfers. The testbench provides structured tasks to automate test execution, including issuing read requests, monitoring the busy and error flags, and comparing the module's computed X, Y, and Z outputs against known expected values. It conducts a comprehensive series of tests (covering initialization, normal reads, edge cases such as all-zeros and all-ones data, alternating bit patterns, back-to-back reads, and error handling via intentional NACK injection) to ensure the robustness of the driver's finite-state machine and data-assembly logic.

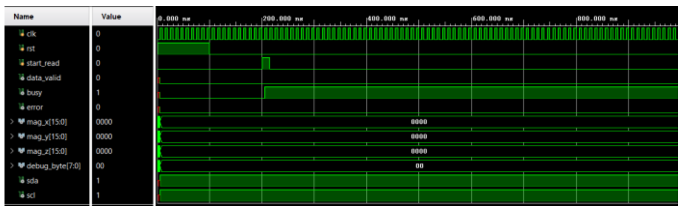
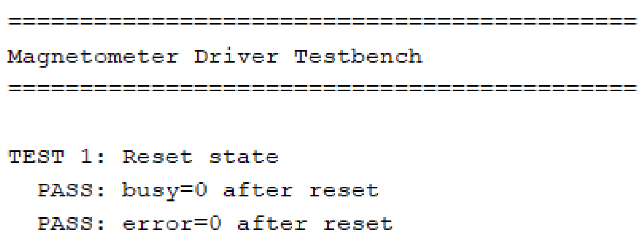


Figure 5. Magnetometer Driver Testbench Simulation Waveform



```
================================================
Magnetometer Driver Testbench
================================================

TEST 1: Reset state
  PASS: busy=0 after reset
  PASS: error=0 after reset
```

Figure 6. Magnetometer Driver Testbench Console Output

The heading_calculation module computes the compass headings from raw magnetometer X and Y readings. It instantiates the tilt_compensation module, which acts as a sensor driver wrapper, to acquire raw magnetometer data and provide basic tilt-compensated outputs, though pitch and roll are disabled for stability in this design. The module converts the magnetometer readings into a 16-direction heading using a quadrant-aware atan2 approximation and predefined sector boundaries to achieve 22.5° resolution. A configurable calibration offset aligns the computed heading with true North, and updates are rate-limited to every 64 samples (~640 ms) to stabilize the display.

The heading_calculation_tb module is designed to verify the functionality of the heading_calculation module without requiring I²C communication with the magnetometer. It operates with a 100 MHz clock and directly injects mock X and Y magnetometer readings to simulate all 16 compass directions. The module performs axis rotation, applies a deadband threshold, determines sign and absolute values, and uses ratio comparisons to assign the heading to one of 16 sectors. A combinational always block calculates the 9-bit heading based on quadrant and sector information, covering cardinal, intercardinal, and secondary intercardinal directions. The testbench also includes a structured run_test task that automates the application of test vectors, compares the computed heading to expected values, and logs pass/fail results with tolerance for angular wrap-around. Finally, the initial block executes tests for all major and intercardinal headings, displays a summary of results, and concludes with a completion message.

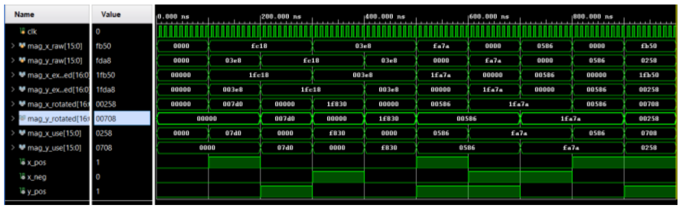
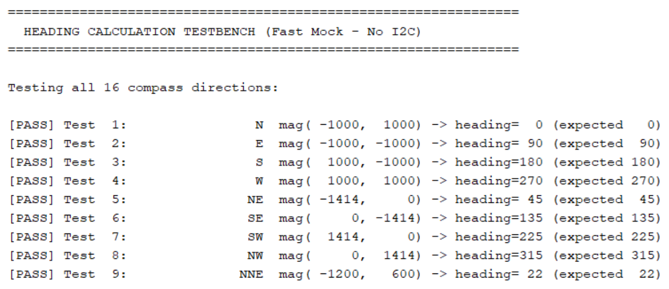


Figure 7. Heading Calculation Testbench Simulation Waveform



```
=================================================================
  HEADING CALCULATION TESTBENCH (Fast Mock - No I2C)
=================================================================

Testing all 16 compass directions:

[PASS] Test  1:                N  mag( -1000,  1000) -> heading=  0 (expected   0)
[PASS] Test  2:                E  mag( -1000, -1000) -> heading= 90 (expected  90)
[PASS] Test  3:                S  mag(  1000, -1000) -> heading=180 (expected 180)
[PASS] Test  4:                W  mag(  1000,  1000) -> heading=270 (expected 270)
[PASS] Test  5:               NE  mag( -1414,     0) -> heading= 45 (expected  45)
[PASS] Test  6:               SE  mag(     0, -1414) -> heading=135 (expected 135)
[PASS] Test  7:               SW  mag(  1414,     0) -> heading=225 (expected 225)
[PASS] Test  8:               NW  mag(     0,  1414) -> heading=315 (expected 315)
[PASS] Test  9:              NNE  mag( -1200,   600) -> heading= 22 (expected  22)
```

Figure 8. Heading Calculation Testbench Console Output

The tilt_compensation module serves as a simplified sensor driver wrapper, providing direct access to raw magnetometer readings without performing tilt compensation to ensure stability. It interfaces with an accelerometer via SPI and a magnetometer via I²C, though accelerometer data is retained for potential future use and not currently applied in heading calculations. The module outputs both raw and "compensated" magnetometer X and Y values (which in this design are identical to the raw data) along with debug signals, error and busy flags, and placeholder pitch and roll values set to zero. It supports simulation by allowing direct injection of test magnetometer data, and includes hardware polling logic to trigger magnetometer reads approximately every 10 ms. Additionally, a data_valid pulse is generated at regular intervals to signal that new magnetometer data is ready for downstream heading computation.

The tilt_compensation_tb module serves as a comprehensive verification environment for the tilt compensation subsystem, enabling controlled testing of magnetometer conversion logic, raw data handling, and output behavior without requiring external SPI or I²C devices. The testbench generates both the 100 MHz system clock and 4 MHz internal clock used by the design, applies a wide range of simulated accelerometer and magnetometer inputs, and evaluates the module's ability to correctly convert unsigned offset-binary sensor readings into signed two's-complement values. Through structured tasks and automated pass/fail tracking, the testbench verifies reset behavior, checks that compensated and raw magnetometer vectors match when tilt correction is disabled, and validates the system's response to cardinal direction simulations and patterned bit sequences. It also confirms proper data-valid pulse timing and ensures that manual test inputs correctly override external sensor data.

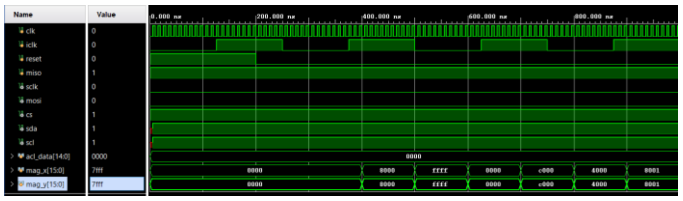
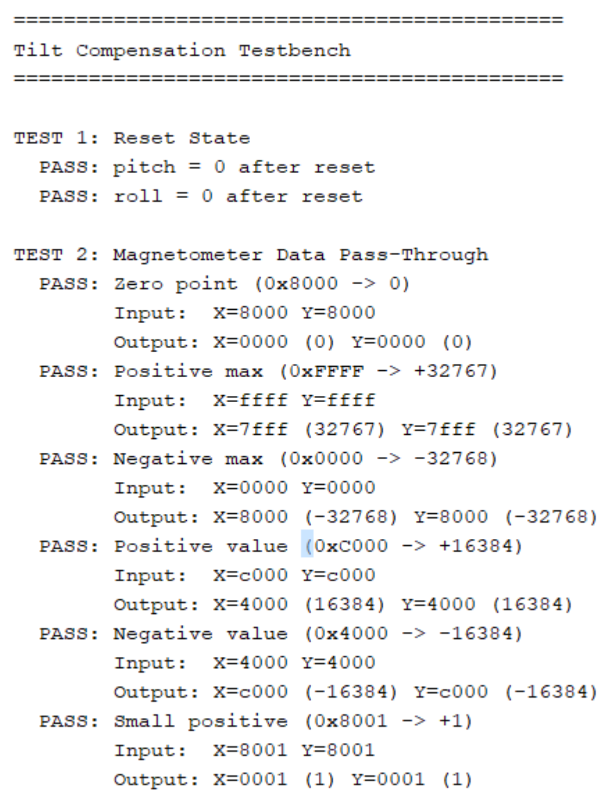


Figure 9. Tilt Compensation Testbench Simulation Waveform



```
============================================
Tilt Compensation Testbench
============================================

TEST 1: Reset State
  PASS: pitch = 0 after reset
  PASS: roll = 0 after reset

TEST 2: Magnetometer Data Pass-Through
  PASS: Zero point (0x8000 -> 0)
        Input:  X=8000 Y=8000
        Output: X=0000 (0) Y=0000 (0)
  PASS: Positive max (0xFFFF -> +32767)
        Input:  X=ffff Y=ffff
        Output: X=7fff (32767) Y=7fff (32767)
  PASS: Negative max (0x0000 -> -32768)
        Input:  X=0000 Y=0000
        Output: X=8000 (-32768) Y=8000 (-32768)
  PASS: Positive value (0xC000 -> +16384)
        Input:  X=c000 Y=c000
        Output: X=4000 (16384) Y=4000 (16384)
  PASS: Negative value (0x4000 -> -16384)
        Input:  X=4000 Y=4000
        Output: X=c000 (-16384) Y=c000 (-16384)
  PASS: Small positive (0x8001 -> +1)
        Input:  X=8001 Y=8001
        Output: X=0001 (1) Y=0001 (1)
```

Figure 10. Tilt Compensation Testbench Console Output

The top module serves as the top-level integration point, connecting the system clock, reset, SPI accelerometer, I²C magnetometer, seven-segment display, and debug LEDs. It generates a 4 MHz SPI clock from the 100 MHz system clock using a clock divider and global BUFG to ensure proper timing for the SPI interface. The module instantiates the heading_calculation submodule, which handles magnetometer readings via I²C, calculates the compass heading, and outputs debug information, with tilt compensation disabled for stability. Enhanced debug LEDs provide real-time system feedback, including heading validity, reset status, I²C busy/error states, and the presence of magnetometer data. The module also converts the 9-bit heading into BCD digits for display on a multiplexed seven-segment display, refreshing each digit for visibility while showing hundreds, tens, and ones positions.

The top testbench provides a full simulation environment for validating the FPGA-based digital compass design, including its heading-calculation logic, I²C magnetometer interface, SPI accelerometer interface, and seven-segment display outputs. It generates a 100 MHz system clock, drives reset, and includes a complete behavioral I²C slave model that emulates the Pmod CMPS2 magnetometer at address 0x30. The model supports register addressing, read/write transactions, start/stop detection, ACK/NACK behavior, and dynamically supplies magnetometer readings based on the testbench's signed simulation inputs. The testbench also exposes internal DUT signals for debugging, such as raw magnetometer values, busy/error indicators, and the computed heading. Automated tasks apply a variety of simulated magnetic field vectors representing 16 compass directions, wait for the heading output to stabilize, and verify correctness within a specified tolerance using a structured pass/fail reporting system. A watchdog ensures the simulation does not hang, and a final summary reports test statistics.

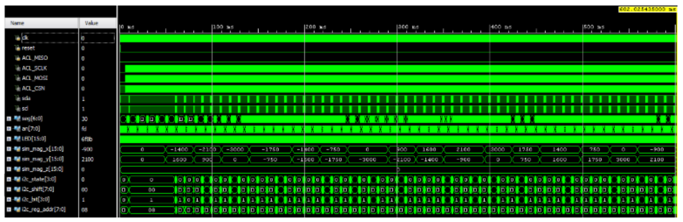


Figure 11. Top Testbench Simulation Waveform

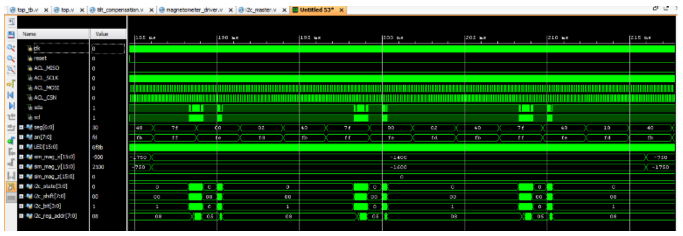


Figure 12. Top Testbench Simulation Waveform for Case East (90 degrees)

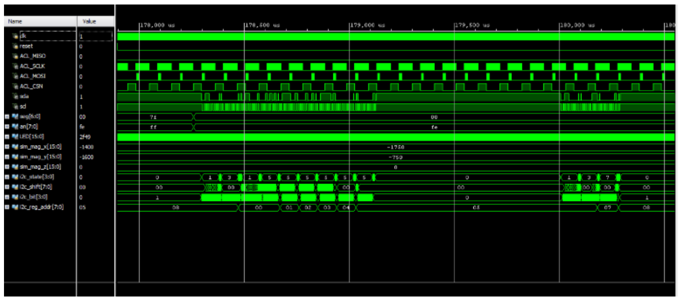


Figure 13. Top Testbench Simulation Waveform for Case East (90 degrees) Zoomed In on SDA Reading

The .xdc file specifies which physical pins on the FPGA are connected to the board's peripherals and sets their electrical standards. The file includes assignments for the system clock, a reset switch, SPI pins for the onboard accelerometer, I^{2}C pins for a magnetometer on the JA header, LEDs for debugging, and the 7-segment display segments and anodes.

IV. EVALUATION

The project was successfully completed as intended. The main goal of implementing a digital compass using the Nexys A7-100T board, integrating an accelerometer (SPI) and magnetometer (I2C), and displaying heading data on 7-segment displays was achieved. All core functionality, including heading calculation, LED debugging outputs, and BCD-to-7-segment conversion, was implemented and verified on hardware.

To verify the project, a combination of simulation and hardware testing techniques was employed. Initially, the Verilog modules were tested using a testbench, which confirmed the correct operation of the binary-to-BCD conversion and 7-segment multiplexing logic. Self-checking testbenches were also utilized in this process. After simulation, the design was synthesized and loaded onto the FPGA. Hardware verification included observing the LED debug outputs to confirm correct data flow, checking the operation of the SPI and I2C interfaces by monitoring activity signals (e.g., mag_busy, mag_error), and confirming that the 7-segment display correctly represented the calculated heading. Additionally, signal stretching techniques were used for visibility of short pulses, ensuring that transient events like heading_valid and mag_error could be reliably observed during testing. These steps ensured both functional correctness and stability of the compass system on the physical board.

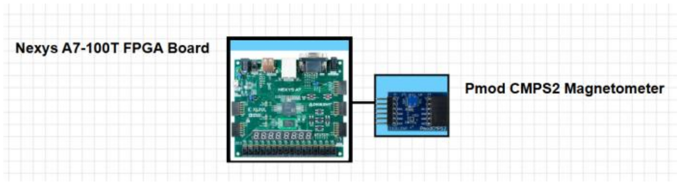


Figure 14. System Diagram

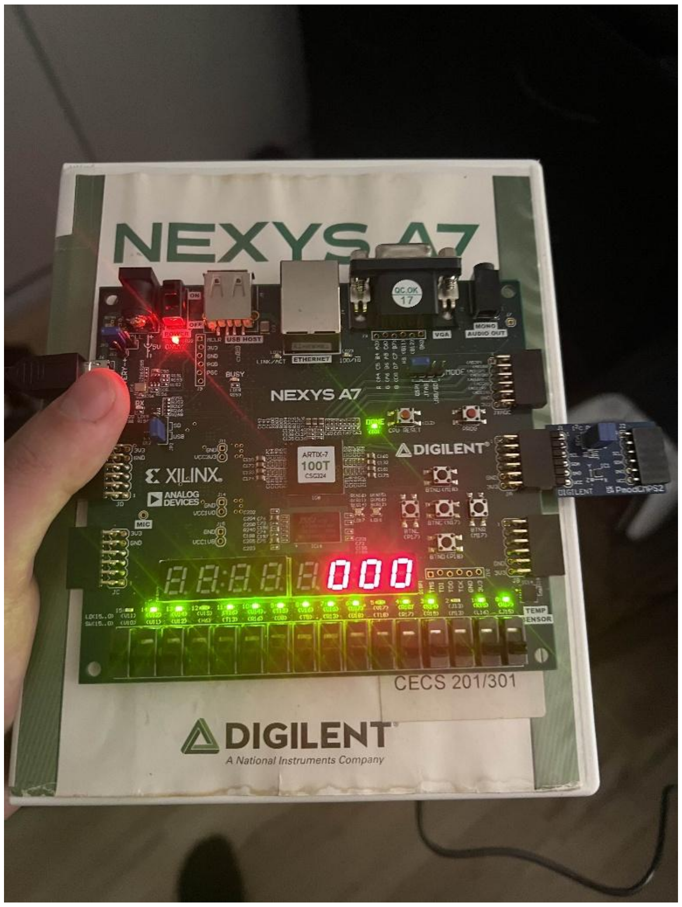


Figure 15. Running Board (North)

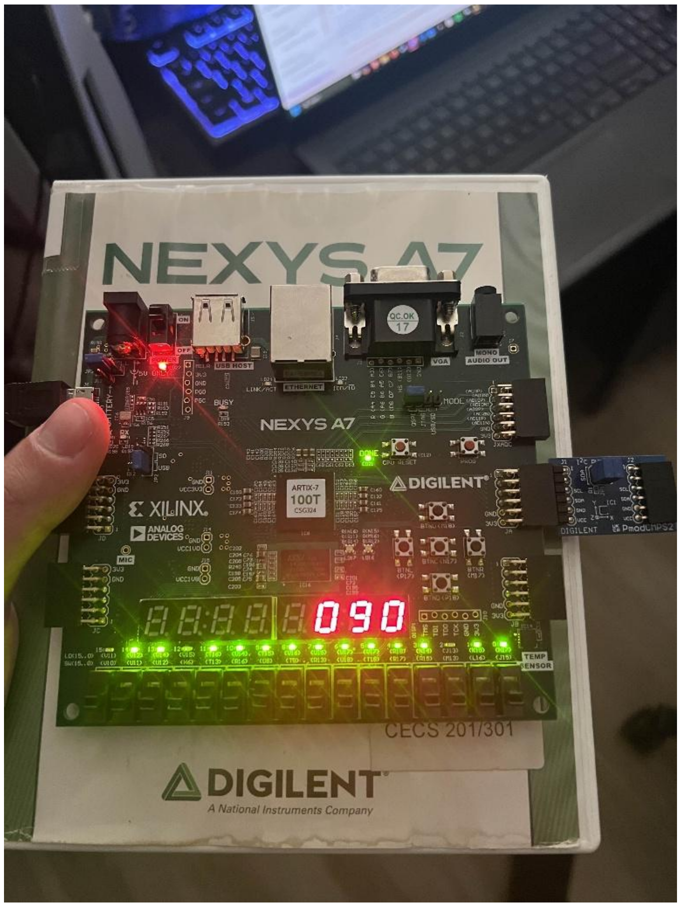


Figure 16. Running Board (East)

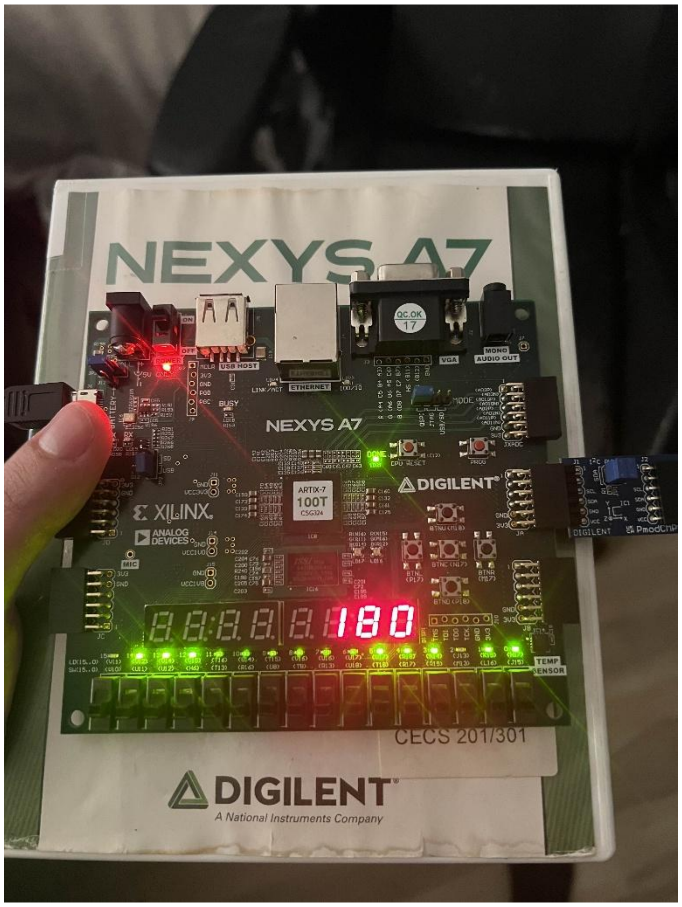


Figure 17. Running Board (South)

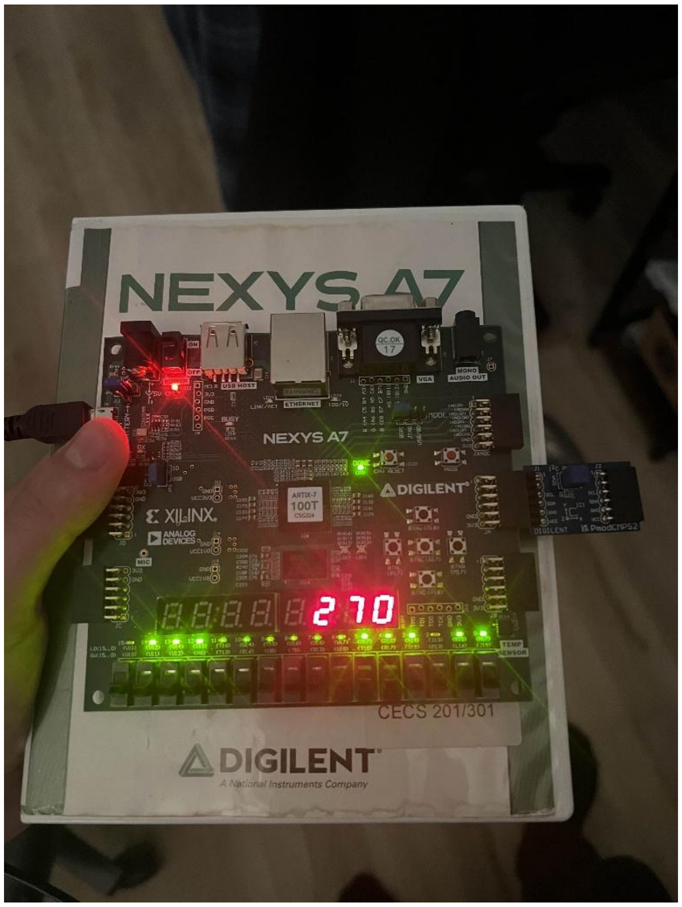


Figure 18. Running Board (West)

V. CONCLUSION

The digital compass project was successfully implemented and verified on the Nexys A7-100T FPGA board. The system demonstrated accurate heading calculations using data from the onboard accelerometer and the Pmod CMPS2 magnetometer. Debug LEDs and 7-segment displays provided clear and reliable feedback for system operation. Through careful simulation, testbench verification, and hardware testing, all modules, including SPI and I2C interfaces, binary-to-BCD conversion, and display multiplexing, were confirmed to function as intended.

This project highlights the effectiveness of combining hardware description languages, peripheral communication protocols, and FPGA-based design to create a functional embedded system. Future work could focus on enhancing the compass accuracy through calibration, expanding the display interface, or integrating additional sensors to create a more robust navigational system.

REFERENCES

[1] Analog Devices, “Data Sheet: ADXL362,” 2023. [Online]. Available: https://www.analog.com/media/en/technical-documentation/data-sheets/adxl362.pdf

[2] Anthropic, “Claude AI, Sonnet 4.5,” [Online]. Available: https://claude.ai

[3] Digilent Inc., “Pmod CMPS2 Reference Manual,” 2017. [Online]. Available: https://digilent.com/reference/pmod/pmodcmps2/reference-manual?srsltid=AfmBOopOPwuOzFculhvFJR94rtt05pmNIkEX_-fWWsLE0fJwwVuUb9OO