Lab 6: Data Acquisition and Logging(Simulation)

Report on Reproducibility and Data Integrity

1. System Overview

A temperature data acquisition system was implemented in Python using the TemperatureDAQSimulator class. The system simulates a temperature sensor, performs real-time monitoring with threshold-based control, and exports data to CSV format with comprehensive integrity features.
Tools Used:

- Python 3.10+ with NumPy and Matplotlib libraries
- VS Code as development environment
- CSV format for data export

2. Data Integrity Implementation

The system implements multiple data integrity mechanisms:

2.1 Timestamping

- Format: ISO 8601 (e.g., 2025-01-16T10:30:45.123Z)
- Precision: Millisecond level
- Purpose: Provides temporal ordering and audit trail

2.2 Sequential Sample Numbering

- Each sample receives a unique index (0, 1, 2, ...)
- Purpose: Detects missing samples and ensures completeness
- Verification: Expected samples = Duration × Sampling Rate

2.3 Data Validation

- Range check: -50°C to 150°C
- NaN detection
- Invalid samples flagged but retained for diagnostics
- Result: All experimental runs achieved 100% data quality

2.4 Checksums

- Generated using: hash(temperature + timestamp)
- 8-character hexadecimal format
- Purpose: Detects data corruption or unauthorized modification
- Verification: All checksums unique across datasets

2.5 Metadata Preservation

Each CSV export includes:

- Session start time
- Total samples collected
- Sampling rate, base temperature, threshold
- Export timestamp
- Total alerts triggered

Purpose: Enables complete session reconstruction and reproducibility

3. Reproducibility Analysis

3.1 Reproducibility Test

Three runs were conducted with slightly different parameters from the first run with configuration below:

- Base Temperature: 28°C
- Threshold: 30°C
- Sampling Rate: 1 Hz
- Duration: 60 seconds

3.2 Assessment

What is Reproducible:

- Statistical properties (mean ≈ base temperature)
- System behavior (oscillation patterns)
- Data structure and format
- Control logic responses

What is Not Reproducible:

- Exact temperature values (due to random noise simulation)
- Precise number of alerts (varies slightly)
- Specific timestamps

Conclusion: The system demonstrates statistical reproducibility appropriate for scientific simulation. Variations between runs reflect realistic sensor behavior (noise and drift), not system flaws. For exact reproducibility, a fixed random seed could be implemented.

4. Data Verification Results

- Completeness Check (Run 1):
- Expected samples: 60
- Actual samples: 60

- Missing samples: 0
- Completeness: 100%

Temporal Integrity:
- All timestamps monotonically increasing
- Average sampling interval: 1.00 seconds (±0.02s)
- Timing accuracy: 99.8%

Checksum Uniqueness:
- Total checksums: 60
- Unique checksums: 60
- Uniqueness: 100%

5. Conclusions

- The Python-based DAQ system successfully implements industry-standard data integrity practices including timestamping, validation, checksums, and metadata preservation. The system demonstrates appropriate reproducibility for simulated data, with consistent statistical properties across runs while maintaining realistic sensor variability. All experimental data showed 100% validity, completeness, and integrity verification.
- Key Finding: Data integrity mechanisms ensure trustworthy data collection suitable for scientific analysis, while documented methodology enables experiment replication with statistically consistent results.

Files Submitted:
- temperature_daq.py (source code)
- 3 CSV data files (run1, run2, run3)
- VI diagrams and flowcharts
- Screenshots of graphs in each run (6 images)