CRC-8 SAE J1850 Implementation for UART Communication

Overview

This project implements a CRC-8 (SAE J1850) algorithm for verifying data integrity in UART communication between two devices. The solution includes CRC calculation, UART message generation, validation, and error detection testing.

Files

1. **CRC_08_LFSR.py** - Implements the CRC-8 calculation and message validation logic.
2. **test_suite.py** - Contains unit tests to verify the correctness of the CRC implementation.
3. **error_modeling_test_suite.py** - Simulates error conditions and evaluates the detection capability of the CRC algorithm.

Dependencies

Ensure you have Python 3 installed. No additional external libraries are required.

Installation

Download the files to your local machine:

```
$ cd <repository_directory>
```

Usage

Running CRC Computation and Validation

One can directly execute ***CRC_08_LFSR.py*** in Python to use the CRC functions in their application.

```
from CRC_08_LFSR import CRC8_SAE_J1850

crc = CRC8_SAE_J1850()
data = bytes([0x12, 0x34, 0x56, 0x78, 0x9A, 0xBC, 0xDE, 0xF0])
uart_message = crc.generate_uart_message_from_device_A(data)
print("Generated UART Message:", uart_message.hex())
valid = crc.validate_received_message_at_device_B(uart_message)
print("Validation Result:", valid)
```

CRC-8 SAE J1850 Implementation for UART Communication

Running Test suite

To validate the correctness of the CRC implementation, run:

```
$ python test_suite.py
```

This executes multiple test cases, including normal operation, single-byte corruption, and boundary condition tests.

Running Error Modeling Tests

To evaluate CRC-8 error detection performance:

```
$ python error_modeling_test_suite.py
```

This script simulates random bit flips in messages and measures CRC detection rates and execution times.

Expected Output

Successful execution of the test suite should output messages indicating whether CRC validation passed or failed for each test case. The error modeling test will display the percentage of detected errors for different corruption scenarios.

Notes

- The UART message includes an 8-byte payload followed by a 1-byte CRC checksum.
- CRC errors are identified and reported with bit-level differences.
- Execution time for CRC-8 calculation is benchmarked in error_modeling_test_suite.py.

License

Author

Manojpriyadharson Kannan