Ahmed Noori, Hussein Sameer

ID: 2210182, 2210047

Instructor: Dr. Hassan Hajjdiab

Date 11/26/2024

**“Implementation of Checksum Algorithm for Error Detection Using MARIE Assembly”**

1. **Introduction:**

In computer networks, the integrity of data transmitted between sender and receiver is crucial. Various factors like noise, interference, or transmission errors can corrupt the data. To ensure error detection, a checksum algorithm is widely used. This project aims to implement a checksum algorithm for error detection in a simulated environment using the MARIE machine architecture and its assembly language. The project is divided into two parts: a detailed explanation of the checksum algorithm and its implementation in MARIE assembly language for both the sender and receiver.

**2. Overview of the Checksum Algorithm**

The checksum algorithm is a simple error-detection technique where:

1. Data bytes are summed together, ignoring any carry bits.
2. The checksum byte is computed as the 2’s complement of the sum.
3. During transmission, the original data bytes and the checksum byte are sent together.
4. At the receiver end, the sum of all received bytes (including the checksum) is calculated:
   * If the sum equals zero, the data is error-free.
   * If the sum is not zero, an error is detected, and retransmission is requested.

This method ensures that single-bit errors or some combinations of multiple-bit errors can be detected.

**3. Implementation in MARIE Assembly Language**

The implementation is divided into two parts: Sender and Receiver. The MARIE assembly language is used to simulate these processes.

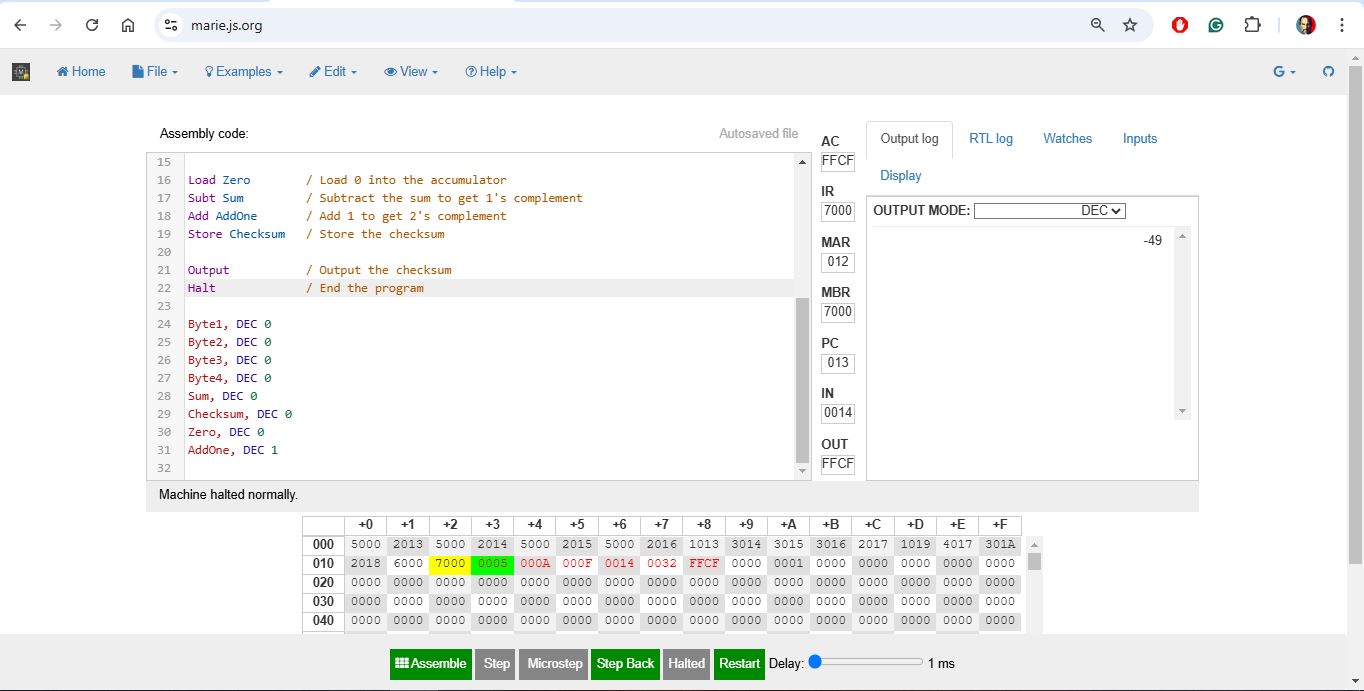
**3.1. Sender Code**

The sender code calculates a **2's complement checksum** for four input bytes. It follows these steps:

**Explanation:**

1. Inputs four data bytes (Byte1, Byte2, Byte3, Byte4).
2. Computes their sum and stores it in Sum.
3. Calculates the **1's complement** by subtracting the sum from 0.
4. Derives the **2's complement** by adding 1 to the 1's complement.
5. Outputs the computed checksum.

The checksum ensures data integrity during transmission by allowing the receiver to detect errors.



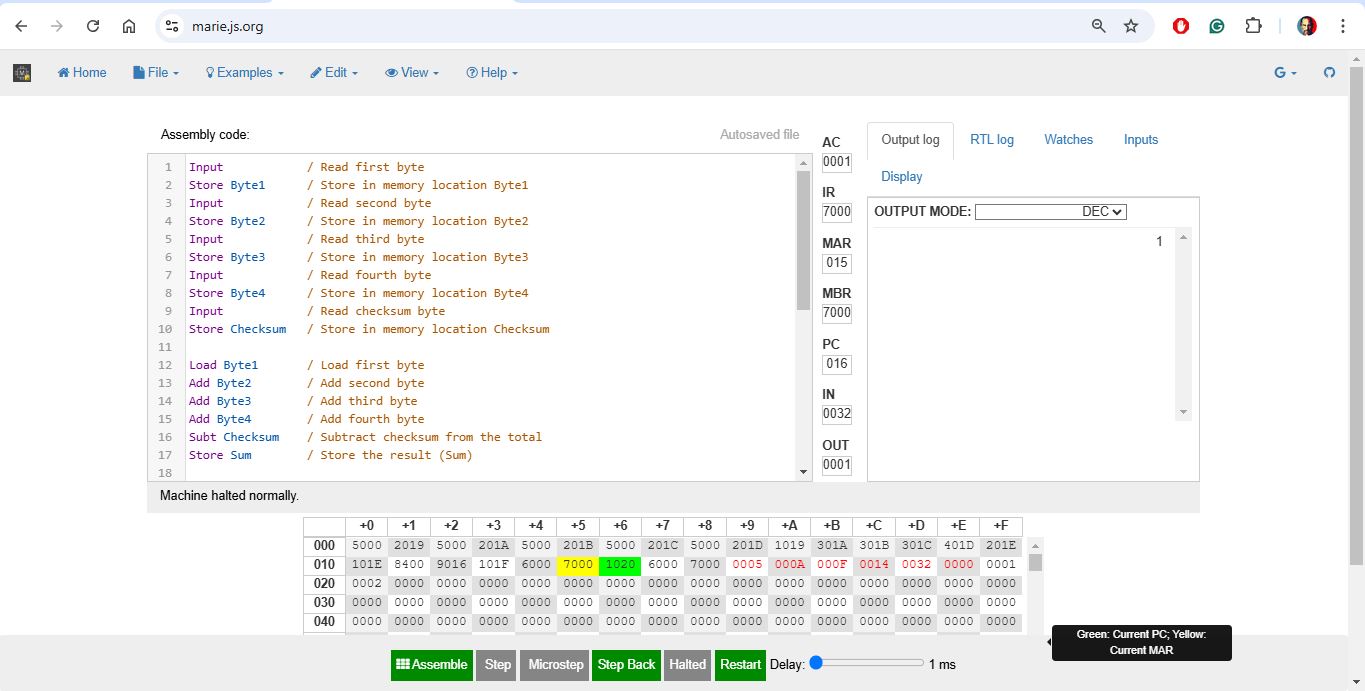
**3.2. Receiver Code**

The receiver code validates the transmitted data by recalculating the sum of the received bytes and the checksum. It works as follows:

**Explanation:**

1. Inputs the four data bytes and the transmitted checksum.
2. Adds the four bytes and the checksum to compute a total sum.
3. Verifies the total sum:
   * If the sum equals 0, the data is valid, and a success message is output.
   * If the sum is non-zero, an error message is output.

This process confirms data integrity and detects transmission errors.



**Key Differences**

****

**4. Testing and Results**

**Byte1 = 5**

**Byte2 = 10**

**Byte3 = 15**

**Byte4 = 20**

**Output from Sender:**

Checksum = 50

Transmitted Data: 5,10,15,20,50

**Sample Input for Receiver:**

Received Data: 5,10,15,20,50

**Output from Receiver:**

Data is valid which is 1 in output box display.

**Case with Error:** If any byte is corrupted during transmission (e.g., Byte3 = 31 instead of 30), the receiver detects the error and requests retransmission.

**5. Challenges and Lessons Learned**

* **Challenges:** Understanding MARIE's limited instruction set and finding alternatives for operations like negation.
* **Lessons Learned:** The project provided valuable insights into error detection techniques, binary arithmetic, and low-level programming using MARIE.

1. **Conclusion**

This project demonstrated the implementation of a checksum algorithm in MARIE assembly language for error detection. The sender and receiver codes work together to ensure data integrity during transmission. The exercise highlighted the importance of error-detection methods in reliable data communication systems.

**References**

 MARIE.js Simulator - <https://marie.js.org/>

 Stallings, W. (2018). Computer Organization and Architecture.

 IEEE Template for Reports -<https://www.ieee.org/conferences/publishing/templates.html>