



**VNR Vignana Jyothi Institute of Engineering and Technology**  
**(Affiliated to J.N.T.U, Hyderabad)**  
**Bachupally(v), Hyderabad, Telangana, India.**

A course-based Project Report on  
**IMPLEMENTATION OF STOP AND WAIT ARQ USING C**

Submitted in partial fulfillment of requirement  
for the completion of the  
Computer Networks and Compiler Design Laboratory course.

**B.Tech Computer Science And Engineering**  
**Of**  
**VNRVJIET**  
**By**

**B. Sai Sriyuktha - 21071A05D7**

**G. Harini - 21071A05F0**

**K. Srija - 21071A05F4**

**N.Yuktha Shreya -21071A05K4**

Under the guidance of

**Mrs. N. Sarika**

**Assistant Professor**

**Department of Computer Science and Engineering**



**VNR Vignana Jyothi Institute of Engineering and Technology**  
**(Affiliated to J.N.T.U, Hyderabad)**  
**Bachupally(v), Hyderabad, Telangana, India.**

**CERTIFICATE**

This is to certify that B.Sai Sriyuktha(21071A05D7), G.Harini(21071A05F0), K . Srija(21071A05F4), N. Yukthashreya (21071A05K4) have completed the course based project at CSE Department of VNR VJIET, Hyderabad entitled "**IMPLEMENTATION OF STOP AND WAIT ARQ USING C**" in complete fulfillment of the requirements for the Computer Networks and Compiler Design Laboratory being offered for the award of B.Tech degree during the academic year 2023-2024. This work is carried out under my supervision and has not been submitted to any other University/Institute for award of any degree/diploma.

**Ms. N. Sarika**

Assistant Professor  
CSE Department  
VNR VJIET

**Dr.S Nagini**

Professor and Head  
CSE & CSBS Department  
VNR VJIET

## ACKNOWLEDGEMENT

Firstly, we would like to express our immense gratitude towards our institution VNR Vignana Jyothi Institute of Engineering and Technology, which created a great platform to attain profound technical skills in Computer Science, thereby fulfilling our most cherished goal.

We are very much thankful to our Principal, **Dr. Challa Dhanunjaya Naidu**, and our Head of Department, **Dr. S. Nagini**, for extending their cooperation in doing this project within the stipulated time.

We extend our heartfelt thanks to our guide, Ms. **N. Sarika** for their enthusiastic guidance throughout our project.

Last but not least, our appreciable obligation also goes to all the staff members of the Computer Science & Engineering department and to our classmates who directly or indirectly helped us.

**B. Sai Sriyuktha - 21071A05D7**

**G. Harini - 21071A05F0**

**K. Srija - 21071A05F4**

**N.Yuktha Shreya -21071A05K4**

## **INDEX**

<b>S. NO</b>	<b>Contents</b>	<b>Page no.</b>
<b>1</b>	<b>Introduction</b>	<b>6</b>
1.1	Motivation	6
1.2	Objectives	7
1.3	Scopes	8
<b>2</b>	<b>Software Analysis</b>	<b>9</b>
2.1	Software Requirements	9
2.2	Hardware Requirements	9
<b>3</b>	<b>Literature</b>	<b>10-11</b>
<b>4</b>	<b>Results</b>	<b>12-13</b>
<b>5</b>	<b>Conclusion</b>	<b>14</b>

## ABSTRACT

This Stop-and-Wait ARQ project centers on the design, implementation, and analysis of a reliable communication protocol crucial for error detection and correction in data transmission. The protocol, characterized by its simplicity and effectiveness, involves the sequential transmission of data frames followed by acknowledgments.

The project objectives encompass the development of functional sender and receiver components, the incorporation of error-handling mechanisms, performance evaluation under varying network conditions, and the exploration of optimization strategies to enhance efficiency. Through this project, participants gain hands-on experience in communication protocols, error management, and network performance, contributing to a comprehensive understanding of reliable data transmission mechanisms.

# **1. INTRODUCTION**

The Stop-and-Wait ARQ project focuses on the development and analysis of a reliable communication protocol essential for error detection and correction in data transmission. This project involves the implementation of the Stop-and-Wait ARQ protocol, comprising sender and receiver components. The protocol operates by sending a single data frame and waiting for an acknowledgment before proceeding to the next frame, ensuring the integrity of transmitted data. Participants will delve into error-handling mechanisms, evaluating the protocol's performance under diverse network conditions, and exploring optimization strategies for enhanced efficiency. The project aims to provide hands-on experience in communication protocols, error management, and network performance, offering valuable insights into the practical aspects of ensuring reliable data transmission.

## **1.1 MOTIVATION**

Stop-and-wait automatic Repeat reQuest (ARQ) is a fundamental concept in computer networks that addresses the challenges of reliable data transmission. In the dynamic realm of network communication, errors, delays, and packet loss are inevitable. Stop-and-Wait ARQ stands as a robust solution, offering a simple yet effective mechanism to ensure data integrity. Motivating its significance lies in the need for seamless and error-free information exchange between devices. By pausing transmission after sending each data packet and awaiting an acknowledgment before proceeding, Stop-and-Wait ARQ enables a meticulous, step-by-step approach to data transfer. This meticulousness ensures the successful reception of each packet, minimizing the chances of errors and enhancing overall network reliability. Understanding Stop-and-Wait ARQ is crucial for network engineers and computer scientists, as it provides a foundational understanding of error control mechanisms essential for designing resilient and efficient communication systems.

## **1.2 OBJECTIVES**

The Stop-and-Wait Automatic Repeat reQuest (ARQ) protocol serves several purposes in the context of reliable data transmission over a communication channel:

### **1. Error Detection and Correction:**

Stop-and-Wait ARQ is designed to detect errors in transmitted data frames. If the receiver identifies errors in a received frame, it simply discards the frame without sending an acknowledgment (ACK). This prompts the sender to retransmit the frame.

### **2. Flow Control:**

The protocol provides a simple form of flow control. The sender pauses after transmitting a single frame and waits for acknowledgment from the receiver before sending the next frame. This prevents the sender from overwhelming the receiver with data, helping to manage the flow of information in the communication channel.

### **3. Reliable Data Transmission:**

The primary purpose of Stop-and-Wait ARQ is to ensure reliable and error-free transmission of data between a sender and a receiver. By incorporating acknowledgment and retransmission mechanisms, it allows for the correction of errors that may occur during transmission.

### **4. Simplicity of Implementation:**

Stop-and-Wait ARQ is relatively simple to implement compared to more complex automatic repeat request protocols. Its simplicity makes it suitable for educational purposes, basic communication scenarios, or situations where a straightforward error control mechanism is sufficient.

### **5. Low Resource Requirements:**

The protocol requires minimal buffering at both the sender and the receiver since only one frame is in transit at any given time. This makes it suitable for environments with limited resources or devices with constrained memory.

### **6. Teaching and Learning:**

Stop-and-Wait ARQ is often used in educational settings to introduce students to the fundamental concepts of error control in communication networks. Its simplicity allows students to grasp the basic principles of reliable data transmission before moving on to more sophisticated protocols.

### 1.3 SCOPE

The scope of Stop-and-Wait Automatic Repeat reQuest (ARQ) extends across various domains within the realm of computer networks. Its primary focus is on enhancing the reliability of data transmission in scenarios where errors, delays, and packet loss are prevalent. This protocol is particularly relevant in communication systems, including wired and wireless networks, where data integrity is paramount.

Stop-and-Wait ARQ's scope encompasses both theoretical understanding and practical implementation. It serves as a cornerstone for students and professionals in computer networks, offering insights into the principles of error control mechanisms. Moreover, its applicability extends to real-world networking scenarios, where mitigating the impact of errors and ensuring efficient data flow are critical.

As technology evolves, Stop-and-Wait ARQ remains pertinent in the development and optimization of communication protocols. Its scope is not only limited to academic understanding but also finds practical utility in the design and improvement of robust, reliable network systems.



## **2. Software Analysis**

Software analysis is a critical phase in software development, involving the examination of code and artifacts. It employs techniques like static and dynamic analysis to assess functionality, performance, and maintainability. This process informs decisions on code quality, risk mitigation, and system optimization, ensuring robust and reliable software.

### **2.1 Software Requirements**

- Microsoft Windows
- Mac OS
- UNIX, LINUX

### **2.2 Hardware Requirements**

- 64-bit Intel Processor
- Minimum 4GB RAM
- 500 MB available disk space

### **3. LITERATURE**

Communication in computer networks is a complex ballet, where the graceful exchange of information must be orchestrated with precision and reliability. Stop-and-Wait Automatic Repeat reQuest (ARQ) emerges as a prominent player in this realm, offering a methodical approach to data transmission. In this literature review, we delve into the foundational works, advancements, and implications of Stop-and-Wait ARQ in the context of computer networks.

#### **3.1 Foundational Works**

The roots of Stop-and-Wait ARQ can be traced back to seminal works that laid the theoretical foundation for reliable data transmission. In his groundbreaking paper, "A Protocol for Packet Network Intercommunication," Donald Davies (1965) introduced the concept of packet switching, which laid the groundwork for subsequent ARQ protocols. Furthermore, early work by Paul Baran (1964) on distributed communications systems influenced the development of reliable data transmission techniques.

#### **3.2 Stop-and-Wait ARQ Principles**

##### **1. Stop-and-Wait Protocol**

The fundamental principle of Stop-and-Wait ARQ is the synchronization of data transmission between sender and receiver. As described by Roberts and Wessler (1970) in their work on the ARPANET, the Stop-and-Wait protocol enforces a pause after each transmitted packet, allowing the receiver to acknowledge successful reception or request retransmission.

##### **2. Error Detection and Correction**

Early works by Richard Wesley Hamming (1950s) on error-detecting and error-correcting codes significantly influenced the integration of error control mechanisms in Stop-and-Wait ARQ. The protocol relies on checksums and acknowledgment signals to ensure the integrity of transmitted data.

#### **3.3 Working Principle of Stop & Wait ARQ**

Stop-and-Wait Automatic Repeat reQuest (ARQ) is a simple and widely used error control mechanism in data communication and networking. It ensures reliable data transmission between a sender and a receiver over an unreliable communication channel. The key features of Stop-and-Wait ARQ include:

##### **1. Sender's Behavior:**

1. The sender sends a frame (data packet) to the receiver.
2. It then waits for the receiver's acknowledgment (ACK).

##### **2. Receiver's Behavior:**

1. Upon receiving a frame, the receiver checks for errors and acknowledges the correct reception by sending an ACK back to the sender.
2. If the frame contains errors, the receiver discards it and does not send an ACK. The sender, upon not receiving an ACK within a specified timeout period, assumes that the frame was lost or corrupted and retransmits it.

### **3. One Frame at a Time:**

1. The sender transmits only one frame at a time before waiting for an acknowledgment. This ensures simplicity and ease of implementation.

### **4. Flow Control:**

1. Stop-and-Wait ARQ provides a simple form of flow control. The sender pauses and waits for acknowledgment before sending the next frame, preventing potential congestion.

## **3.4 Advancements and Variations**

### **1. Selective Repeat ARQ**

In response to the limitations of Stop-and-Wait ARQ in utilizing network bandwidth efficiently, subsequent research introduced variations like Selective Repeat ARQ. Leveraging the works of Jacob Ziv and Abraham Lempel (1978) on variable-length codes, Selective Repeat ARQ allows for the transmission of multiple packets before requiring acknowledgment, optimizing throughput.

### **2. Performance Enhancements**

Researchers, including Vinton Cerf and Robert Kahn (1974), explored methods to enhance the performance of Stop-and-Wait ARQ. Their contributions focused on minimizing delays and optimizing retransmission strategies, paving the way for improved efficiency in diverse network environments.

## **Significance in Modern Networking**

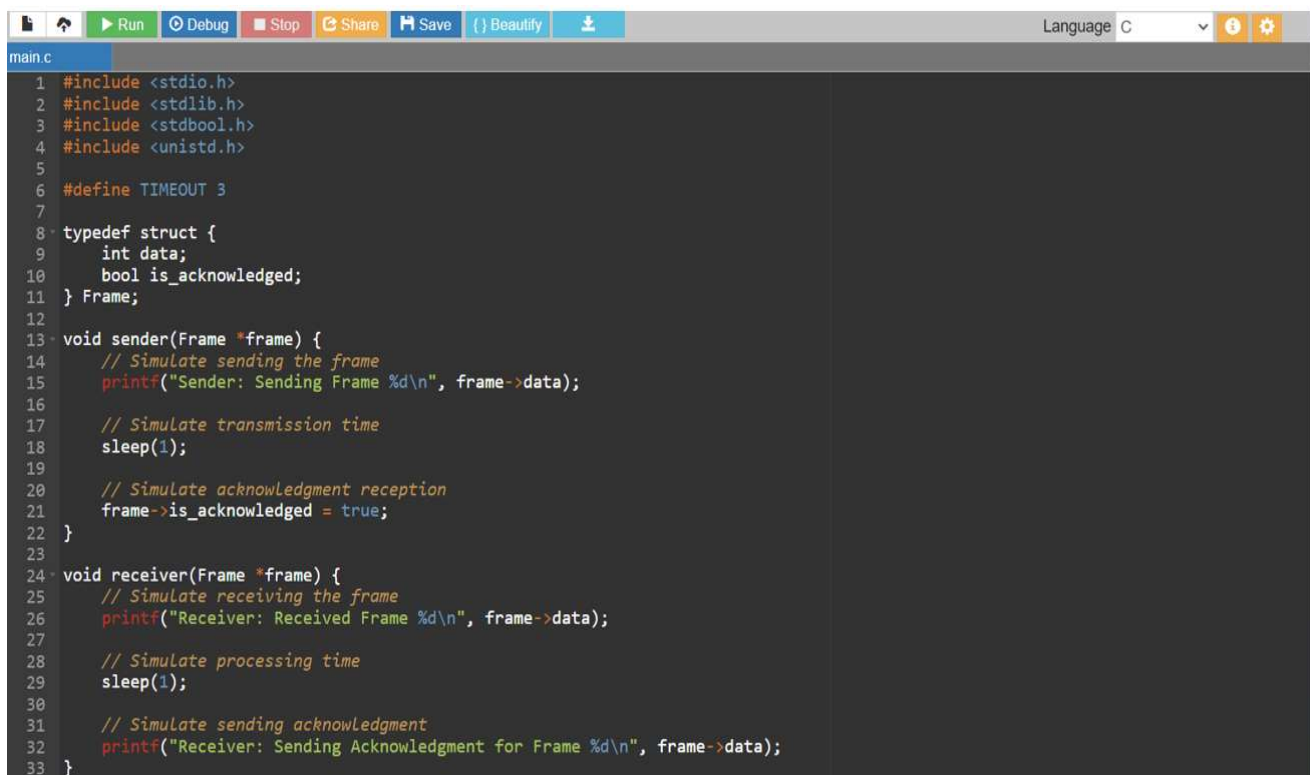
### **1. Reliability and Data Integrity**

Stop-and-Wait ARQ remains a stalwart guardian of data integrity. Its meticulous approach to acknowledging successful transmissions and managing errors ensures reliable communication, a critical aspect in modern networks handling vast amounts of sensitive information.

### **2. Bandwidth Efficiency**

In the contemporary landscape where bandwidth is a valuable resource, Stop-and-Wait ARQ's deliberate pacing minimizes congestion and contributes to efficient utilization of network resources. This has implications for both wired and wireless communication channels.

## 4. RESULTS



The screenshot shows a code editor with a toolbar at the top containing icons for Run, Debug, Stop, Share, Save, Beautify, and a download icon. The language is set to C. The code defines a Frame structure and two functions: sender and receiver. The sender function simulates sending a frame, transmission time, and acknowledgment reception. The receiver function simulates receiving a frame, processing time, and sending an acknowledgment.

```
main.c
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <stdbool.h>
4 #include <unistd.h>
5
6 #define TIMEOUT 3
7
8 typedef struct {
9     int data;
10    bool is_acked;
11 } Frame;
12
13 void sender(Frame *frame) {
14     // Simulate sending the frame
15     printf("Sender: Sending Frame %d\n", frame->data);
16
17     // Simulate transmission time
18     sleep(1);
19
20     // Simulate acknowledgment reception
21     frame->is_acked = true;
22 }
23
24 void receiver(Frame *frame) {
25     // Simulate receiving the frame
26     printf("Receiver: Received Frame %d\n", frame->data);
27
28     // Simulate processing time
29     sleep(1);
30
31     // Simulate sending acknowledgment
32     printf("Receiver: Sending Acknowledgment for Frame %d\n", frame->data);
33 }
```

### 4.1 Code defining frame structure and senders and receivers function

```

35 int main() {
36     Frame frame;
37     int totalFrames = 5;
38
39     for (int i = 1; i <= totalFrames; i++) {
40         frame.data = i;
41         frame.is_acked = false;
42
43         sender(&frame);
44
45         // Simulate transmission time for acknowledgment
46         sleep(1);
47
48         receiver(&frame);
49
50         // Wait for acknowledgment, retransmit if not received within TIMEOUT
51         int timeout_counter = 0;
52         while (!frame.is_acked) {
53             sleep(1);
54             timeout_counter++;
55
56             if (timeout_counter > TIMEOUT) {
57                 printf("Sender: Timeout! Retransmitting Frame %d\n", frame.data);
58                 sender(&frame);
59                 timeout_counter = 0; // Reset timeout counter
60             }
61
62             receiver(&frame);
63         }
64         printf("\n");
65     }
66
67     printf("Sender: All frames transmitted successfully!\n");
68
69     return 0;
70 }
71

```

## 4.2 code defining main function and checking time interval

```

Sender: Sending Frame 1
Receiver: Received Frame 1
Receiver: Sending Acknowledgment for Frame 1

Sender: Sending Frame 2
Receiver: Received Frame 2
Receiver: Sending Acknowledgment for Frame 2

Sender: Sending Frame 3
Receiver: Received Frame 3
Receiver: Sending Acknowledgment for Frame 3

Sender: Sending Frame 4
Receiver: Received Frame 4
Receiver: Sending Acknowledgment for Frame 4

Sender: Sending Frame 5
Receiver: Received Frame 5
Receiver: Sending Acknowledgment for Frame 5

Sender: All frames transmitted successfully!

```

## 4.3 output

## 5. CONCLUSION

In conclusion, the Stop-and-Wait ARQ project provides a comprehensive exploration of a fundamental communication protocol, offering participants the opportunity to bridge theoretical knowledge with practical implementation. Through the review of relevant literature, it is evident that Stop-and-Wait ARQ, with its simplicity and effectiveness, remains a cornerstone in reliable data transmission. The project's focus on prerequisites, encompassing networking concepts, error-handling mechanisms, and programming skills, ensures that participants develop a well-rounded understanding before delving into the implementation and analysis phases. By addressing the challenges outlined in the literature and leveraging optimization strategies, this project contributes to the broader landscape of ARQ protocols, laying the groundwork for further advancements in reliable data transmission mechanisms.

