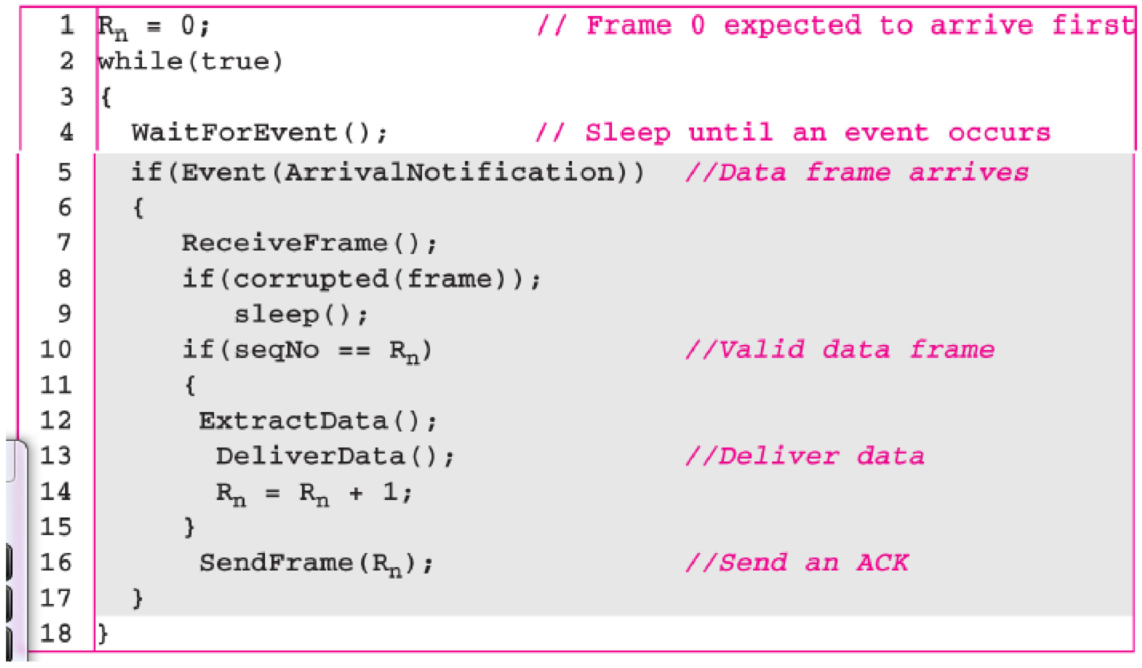
Receiver-site algorithm for Stop-and-Wait ARQ Protocol



```
$R_n$ = 0;                       // Frame 0 expected to arrive first
while(true)
{
  WaitForEvent();                // Sleep until an event occurs
  if(Event(ArrivalNotification))   //Data frame arrives
  {
     ReceiveFrame();
     if(corrupted(frame));
        sleep();
     if(seqNo == $R_n$)              //Valid data frame
     {
      ExtractData();
       DeliverData();                //Deliver data
       $R_n$ = $R_n$ + 1;
     }
      SendFrame($R_n$);                //Send an ACK
  }
}
```

CLIENT SIDE

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <arpa/inet.h>
#include <sys/socket.h>

typedef struct packet{
   char data[1024];
}Packet;

typedef struct frame{
   int frame_kind; //ACK:0, SEQ:1 FIN:2
   int sq_no;
   int ack;
   Packet packet;
}Frame;

int main(int argc, char **argv[]){
   if (argc != 2){
                  printf("Usage: %s <port>", argv[0]);
                  exit(0);
         }

         int port = atoi(argv[1]);
```

```
	int sockfd;
	struct sockaddr_in serverAddr;
	char buffer[1024];
	socklen_t addr_size;

	int frame_id = 0;
	Frame frame_send;
	Frame frame_recv;
	int ack_recv = 1;

	sockfd = socket(AF_INET, SOCK_DGRAM, 0);

	memset(&serverAddr, '\0', sizeof(serverAddr));
	serverAddr.sin_family = AF_INET;
	serverAddr.sin_port = htons(port);
	serverAddr.sin_addr.s_addr = inet_addr("127.0.0.1");

	while(1){

		if(ack_recv == 1){
			frame_send.sq_no = frame_id;
			frame_send.frame_kind = 1;
			frame_send.ack = 0;

			printf("Enter Data: ");
			scanf("%s", buffer);
			strcpy(frame_send.packet.data, buffer);
			sendto(sockfd, &frame_send, sizeof(Frame), 0, (struct
sockaddr*)&serverAddr, sizeof(serverAddr));
			printf("[+]Frame Send\n");
		}
		int addr_size = sizeof(serverAddr);
		int f_recv_size = recvfrom(sockfd, &frame_recv, sizeof(frame_recv), 0
,(struct sockaddr*)&serverAddr, &addr_size);

		if( f_recv_size > 0 && frame_recv.sq_no == 0 && frame_recv.ack ==
frame_id+1){
			printf("[+]Ack Received\n");
			ack_recv = 1;
		}else{
			printf("[-]Ack Not Received\n");
			ack_recv = 0;
		}
		frame_id++;
	}
	close(sockfd);
	return 0;
}
```

Server Side

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <sys/socket.h>
#include <unistd.h>
#include <arpa/inet.h>


typedef struct packet{
    char data[1024];
}Packet;

typedef struct frame{
    int frame_kind; //ACK:0, SEQ:1 FIN:2
    int sq_no;
    int ack;
    Packet packet;
}Frame;

int main(int argc, char** argv){

        if (argc != 2){
                printf("Usage: %s <port>", argv[0]);
                exit(0);
        }

        int port = atoi(argv[1]);
        int sockfd;
        struct sockaddr_in serverAddr, newAddr;
        char buffer[1024];
        socklen_t addr_size;

        int frame_id=0;
        Frame frame_recv;
        Frame frame_send;

        sockfd = socket(AF_INET, SOCK_DGRAM, 0);

        memset(&serverAddr, '\0', sizeof(serverAddr));
        serverAddr.sin_family = AF_INET;
        serverAddr.sin_port = htons(port);
        serverAddr.sin_addr.s_addr = inet_addr("127.0.0.1");

        bind(sockfd, (struct sockaddr*)&serverAddr, sizeof(serverAddr));
```

```
	addr_size = sizeof(newAddr);

	while(1){
		int f_recv_size = recvfrom(sockfd, &frame_recv, sizeof(Frame), 0, (struct 
sockaddr*)&newAddr, &addr_size);
		if (f_recv_size > 0 && frame_recv.frame_kind == 1 && frame_recv.sq_no == 
frame_id){
			printf("[+]Frame Received: %s\n", frame_recv.packet.data);

			frame_send.sq_no = 0;
			frame_send.frame_kind = 0;
			frame_send.ack = frame_recv.sq_no + 1;
			sendto(sockfd, &frame_send, sizeof(frame_send), 0, (struct 
sockaddr*)&newAddr, addr_size);
			printf("[+]Ack Send\n");
		}else{
			printf("[+]Frame Not Received\n");
		}
		frame_id++;
	}

	close(sockfd);
	return 0;
}
```

Output
Client side

```
gcc client.c -o c

./c 4000
Enter Data: 1234
[+]Frame Send
[+]Ack Received
Enter Data: 0100
[+]Frame Send
[+]Ack Received
Enter Data: -5
[+]Frame Send
[+]Ack Received
Enter Data: abc
[+]Frame Send
[+]Ack Received
```

Server side

```
gcc server.c -o s
net@inlab:~/Desktop$ ./s 4000
[+]Frame Received: 1234
```

```
[+]Ack Send
[+]Frame Received: 0100
[+]Ack Send
[+]Frame Received: -5
[+]Ack Send
[+]Frame Received: abc
[+]Ack Send
```