Error Detection using Cyclic Redundancy Code (CRC-8)

Assignment Description

In this application, your aim will be to implement a simple Stop-and-Wait based data link layer level logical channel between two nodes A and B using socket API, where node A and node B are the client and the server for the socket interface respectively. Data link layer protocol should provide the following Error handling technique in Data Link Layer.

- Error Detection using Cyclic Redundancy Code (using CRC-8 as generator polynomial, i.e. $G(x) = x^8 + x^2 + x + 1$)

Operation to Implement:

1. Client should construct the message to be transmitted ($T(x)$) from the raw message using CRC.
2. At the sender side $T(x)$ is completely divisible by $G(x)$ (means no error), send ACK to the sender, otherwise (means error), send NACK to the sender.
3. You must write error generating codes based on a user given BER or probability (random number between 0 and 1) to insert error into both $T(x)$ and ACK/NACK.
4. If NACK is received by the sender, it should retransmit the $T(x)$ again following the above steps.
5. In the client side also implement Timer Mechanism to detect the timeout (in case of error in ACK/ NACK) and retransmit the message $T(x)$ again once time out happens.

You also require implementing a "Concurrent Server", i.e., a server that accepts connections from multiple clients and serves all of them concurrently.

You should accept the IP Address and Port number from the command line (Don't use a hard-coded port number).

Prototype for command line is as follows:

Prototypes for Client and Server
Client: `<executable code> <Server IP Address> <Server Port number>`
Server: `<executable code> <Server Port number>`

The connection to the server should be gracefully terminated. When the server is terminated by pressing Control+C, the server should also gracefully release the open socket (Hint: requires use of a signal handler).