Imagine that you have a microcontroller that communicates to a generic system that may consist of several other boards via UART. How do you ensure that each message is properly sent and that its content is correct?

To ensure that the message is sent correctly, it is necessary to have good reliability in communication, such as:

- **Configuration between transmitter and receiver:** The transmitter and receiver must have the same settings for baud rate, number of bits, stop bit, parity bit (or no parity), and flow control.
- **Baud rate configuration:** The higher the transmission rate, the more robust the hardware needs to be, which can result in data loss. Both transmitter and receiver need to operate at the same transfer rate.
- **Hardware:** In physical terms, the transmitter and the receiver need to have the same high and low-level voltage and share the same reference (ground).

To ensure that the content of the sent message is correct, one can implement some error-checking strategies, such as:

- **Parity Bit:** The parity bit checks for bit inversion caused by electromagnetic interference, but it's not very efficient. If an even number of bit inversions occur, or if the parity bit (which is sent in the same data frame as the message) undergoes a bit inversion, this technique fails to detect that a change has occurred in the data of the sent message.
- **Checksum:** The transmitter calculates a binary sum of the bytes in the message to be sent and than makes complement of set, and generates a result called the checksum. This checksum is sent along with the message. The receiver also calculates a binary sum of the bytes in the received message along with the checksum. If the result of the sum is equal to zero, no content alteration occurred during transmission.
- **CRC (Cyclic Redundancy Check)**: CRC is a more robust method than checksum. CRC works by generating a CRC polynomial from the data and then dividing this polynomial by a fixed polynomial (the generator polynomial) using modular arithmetic. The result of the division is the CRC, which is appended to the data and sent along with it. The receiver performs the same calculation and checks if the received CRC matches the recalculated CRC. If there is a difference, it suggests that the data may have been corrupted.
- **ACK/NACK (Acknowledgment/Not Acknowledgment)**: After the receiver verifies the integrity of the received data, it sends an ACK back to the transmitter if the message is correct. If the message is incorrect or corrupted, the receiver sends a NACK, requesting the transmitter to retransmit the message.
- **Timeouts and Retransmissions:** Timeouts are used by the transmitter to check if the receiver has received the sent message. If the transmitter does not receive an ACK or NACK within a predetermined timeout period, it is understood by the transmitter that the message was lost or corrupted, prompting the transmitter to resend the message.