**UART Communication :**

UART, which stands for Universal Asynchronous Receiver/Transmitter, is a widely used hardware communication protocol that allows for serial communication between devices. It's a simple and reliable way for microcontrollers, computers, and other digital devices to exchange data.

brief overview of UART communication:

**1**. **Hardware Configuration**:

- UART communication typically involves two pins or wires: TX (Transmit) and RX (Receive). These pins are used for sending and receiving data.

- UART operates in a point-to-point or multi-point configuration, where one transmitter communicates with one or more receivers.

**2**. **Data Frame**:

- UART transmits data in the form of a frame, which includes the following components:

- Start bit: Indicates the beginning of the data frame(0/low).

- Data bits (usually 8 bits but can vary): The actual data to be transmitted.

- Parity bit (optional): Used for error checking.

- Stop bits (usually 1 or 2): Indicates the end of the data frame(1/high).

**3. Baud Rate:**

- Baud rate refers to the speed at which data is transmitted over the UART connection and is expressed in bits per second (bps). Common baud rates include 9600, 19200, 115200, etc. Both the transmitting and receiving devices

must operate at the same baud rate for successful communication.

**4. Asynchronous Communication:**

- UART is asynchronous, meaning there is no shared clock signal between the transmitting and receiving devices.

- Instead, both devices agree on a common baud rate, and the receiving device samples the incoming data at regular intervals based on this rate.

**5. Data Transmission:**

- When data transmission begins, the UART sender (transmitter) places the start bit on the TX line, followed by the data bits (and optional parity bit), and then the stop bit(s).

- The receiving UART detects the start bit, samples the data bits, checks the parity (if used), and waits for the stop bit(s) to confirm the end of the frame.

**6. Flow Control:**

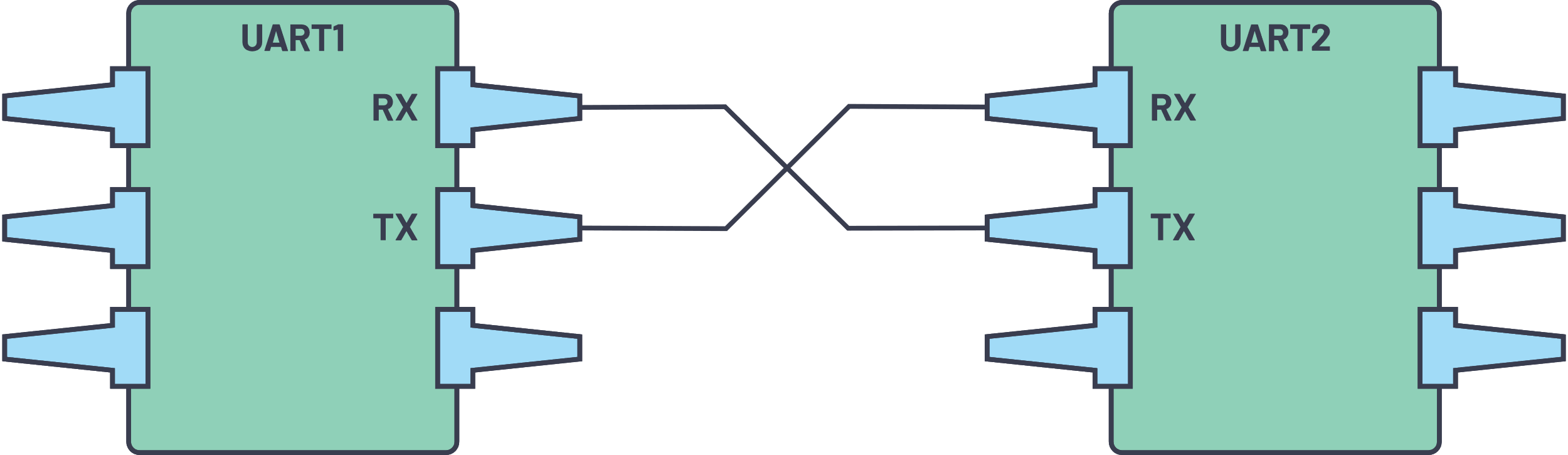
- UART communication doesn't inherently provide flow control mechanisms like hardware (RTS/CTS) or software (XON/XOFF) flow control.

- Flow control may need to be implemented separately if data overrun is a concern.

**7. Error Handling:**

- UART communication can be susceptible to errors, such as noise or data corruption. Parity bit checking can help detect some errors, but it's not foolproof.

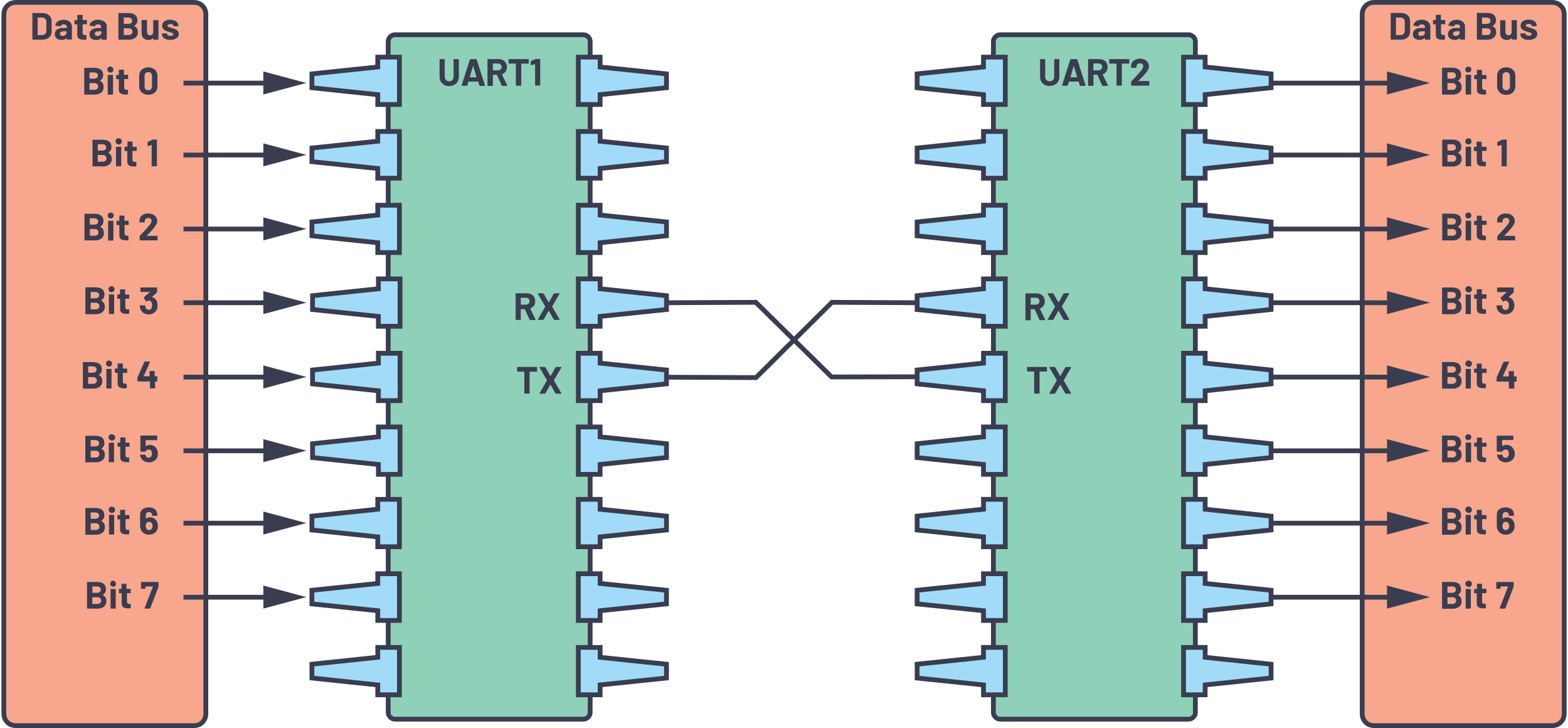
- For more robust error handling, additional protocols like UART with error-checking (e.g., UART with CRC) or higher-layer protocols (e.g., protocols built on top of UART like Modbus or UART-over-USB) are used.

UART communication is widely used in various applications, including serial console communication, interfacing with sensors and peripherals, and connecting microcontrollers or embedded systems to computers or other devices. It's a versatile and relatively simple way to exchange data serially between devices. ****

The two signals of each UART device are named:

* Transmitter (Tx)
* Receiver (Rx)

UART with Data Bus :



UART Packet :



* **Start Bit** :

The UART data transmission line is normally held at a high voltage level when it’s not transmitting data. To start the transfer of data, the transmitting UART pulls the transmission line from high to low for one (1) clock cycle. When the receiving UART detects the high-to-low voltage transition, it begins reading the bits in the data frame at the frequency of the baud rate.

* **Data Frame**:

The data frame contains the actual data being transferred. It can be five (5) bits up to eight (8) bits long if a parity bit is used. If no parity bit is used, the data frame can be nine (9) bits long. In most cases, the data is sent with the least significant bit first.

* **Parity :**

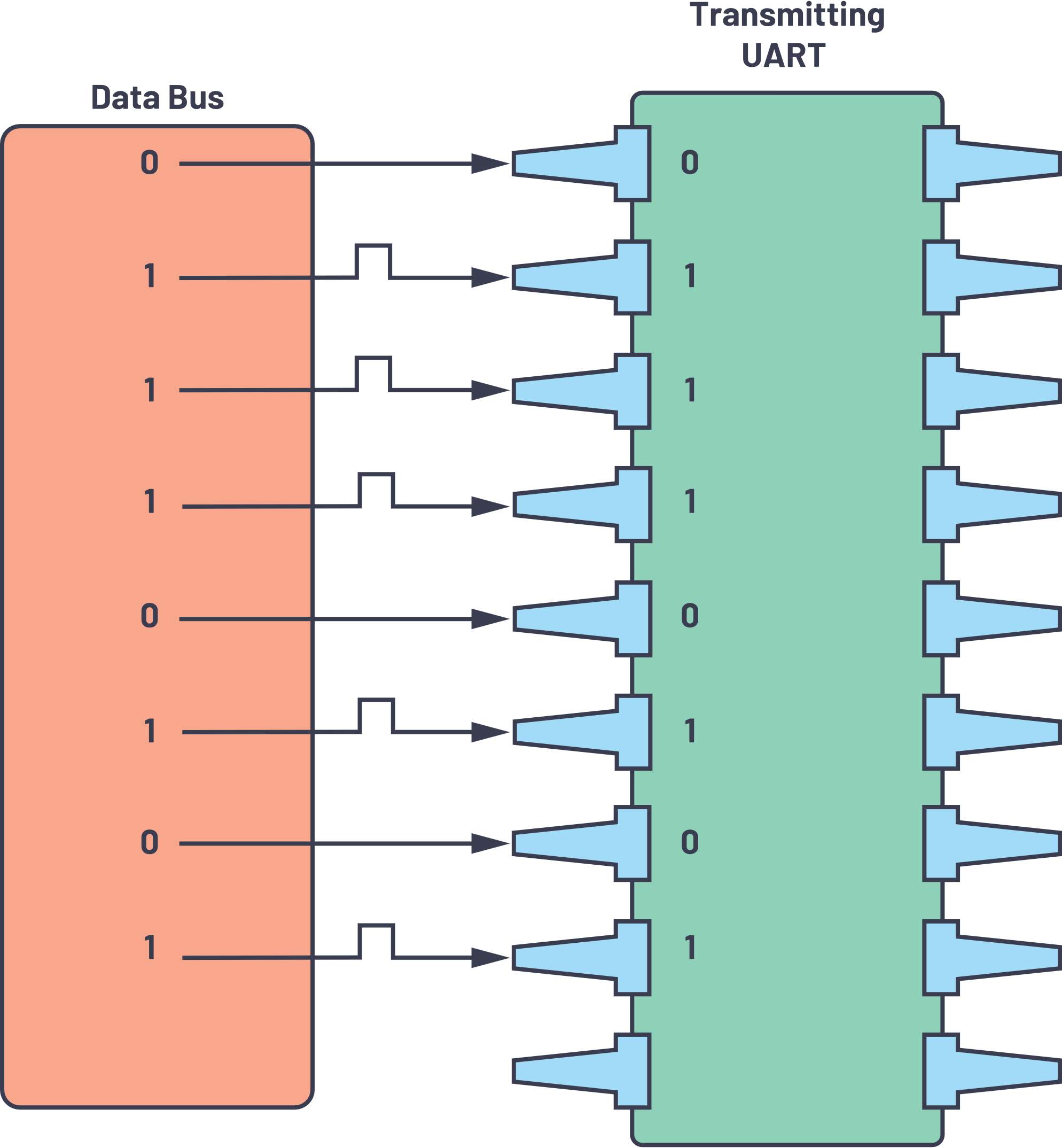
After the receiving UART reads the data frame, it counts the number of bits with a value of 1 and checks if the total is an even or odd number. If the parity bit is a 0 (even parity), the 1 or logic-high bit in the data frame should total to an even number. If the parity bit is a 1 (odd parity), the 1 bit or logic highs in the data frame should total to an odd number.

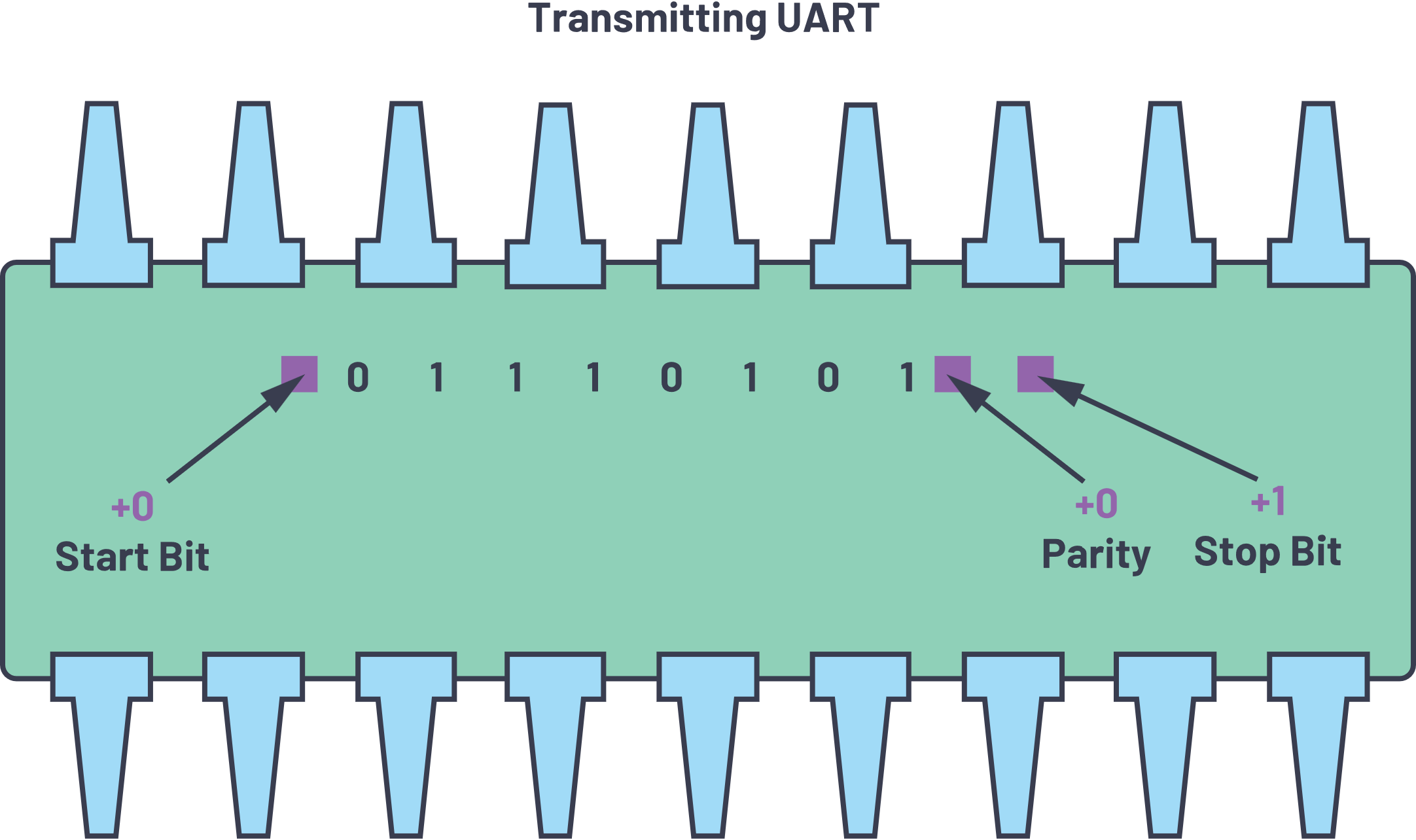
* **Stop Bits:**

To signal the end of the data packet, the sending UART drives the data transmission line from a low voltage to a high voltage for one (1) to two (2) bit(s) duration.

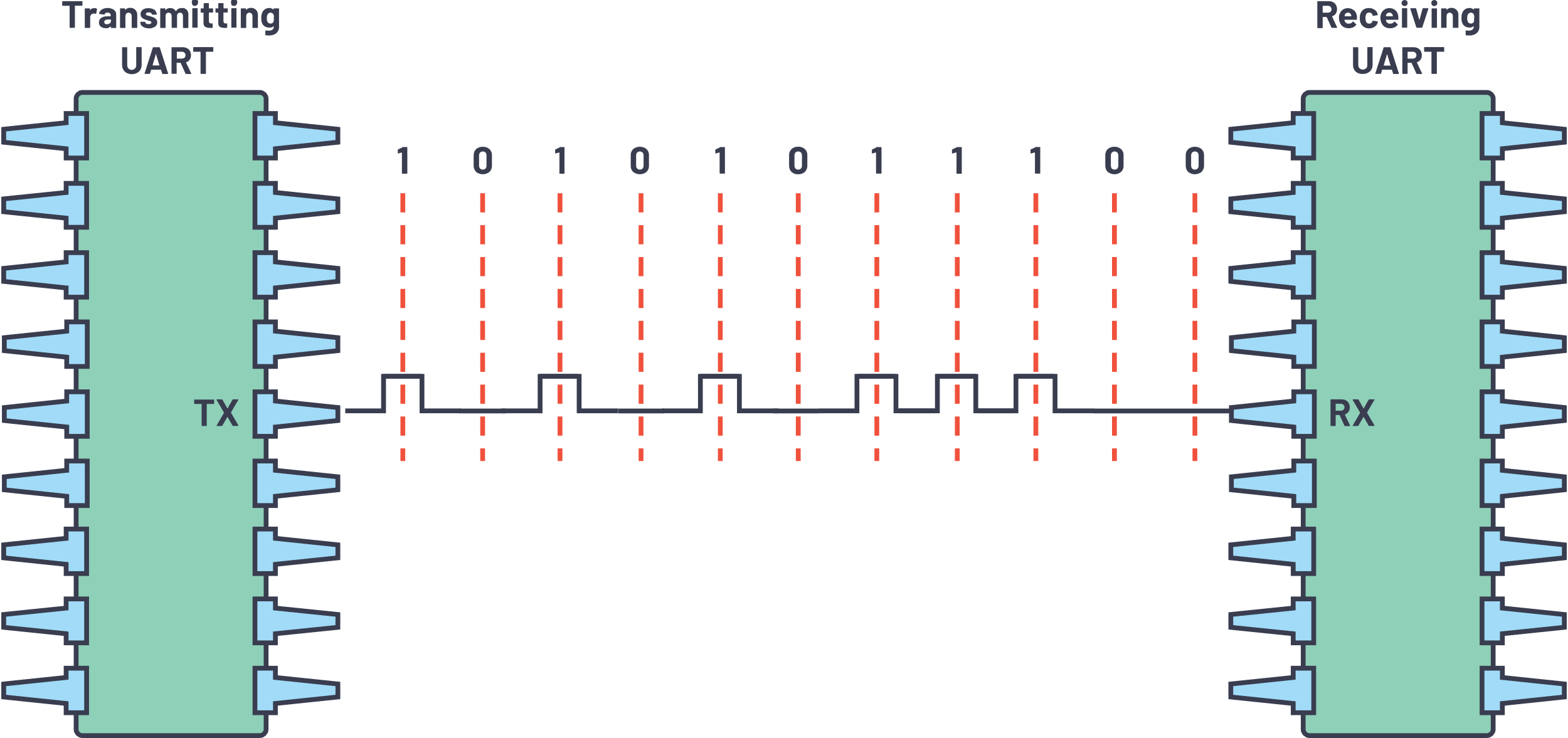
**STEPS OF UART COMMUNICATION**

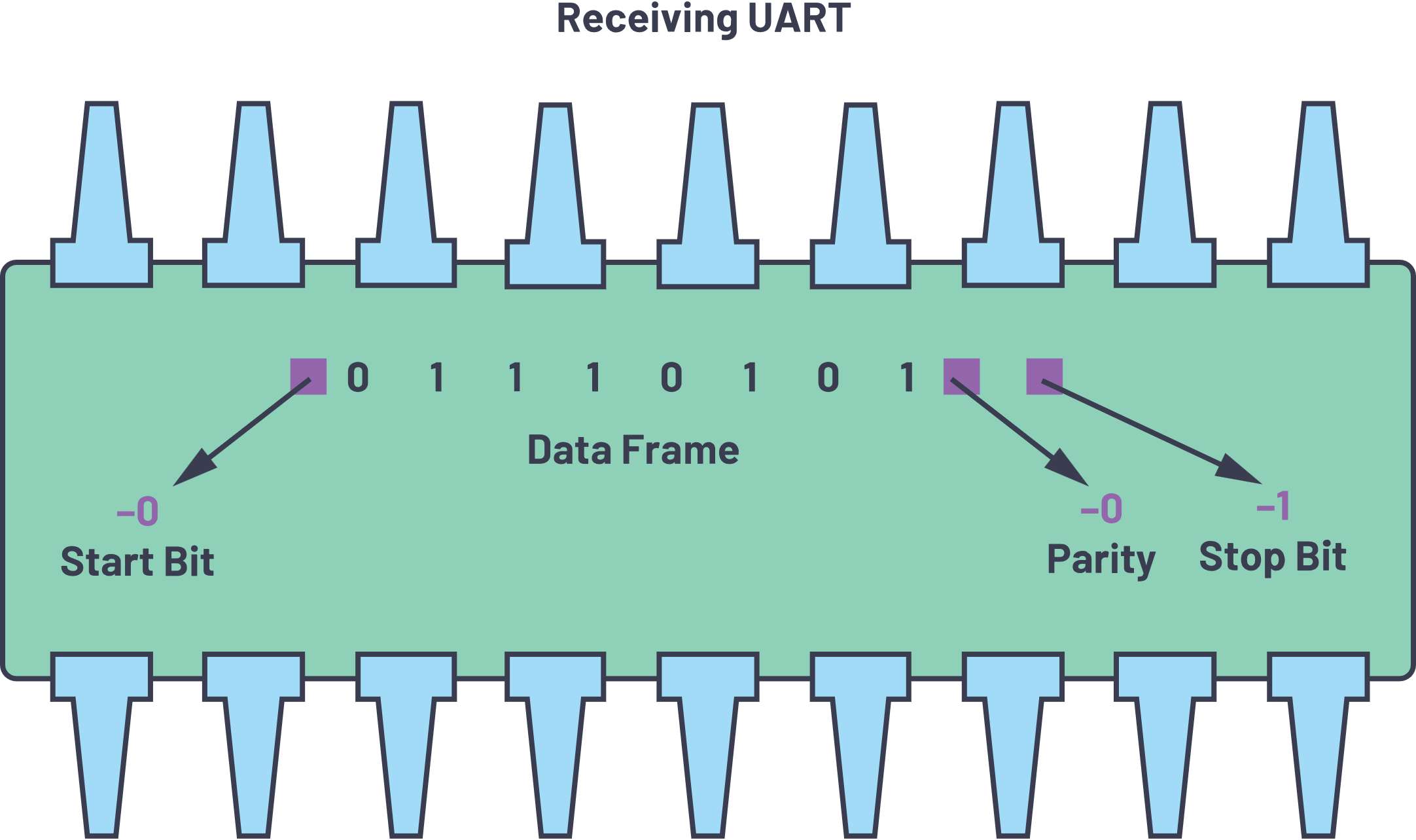
First: The transmitting UART receives data in parallel from the data bus.

****

Second: The transmitting UART adds the start bit, parity bit, and the stop bit(s) to the data frame. 

Third: The entire packet is sent serially starting from start bit to stop bit from the transmitting UART to the receiving UART. The receiving UART samples the data line at the preconfigured baud rate.



Fourth: The receiving UART discards the start bit, parity bit, and stop bit from the data frame. 

Fifth: The receiving UART converts the serial data back into parallel and transfers it to the data bus on the receiving end.

