

1. Summary of Connection Techniques

Microcontrollers generally use three main serial communication protocols to connect with each other. Below is a summary of the techniques suitable for peer-to-peer connections.

UART (Universal Asynchronous Receiver-Transmitter)

- **Wiring:** Requires 2 signal wires (TX and RX) plus a common Ground. It is an asynchronous connection (no clock line).
- **Advantages:** It is the simplest to implement for point-to-point connections. It excels over longer cable distances compared to I2C/SPI and is available on almost all microcontrollers.
- **Disadvantages:** It has lower maximum data speeds and is strictly limited to one-to-one communication. Both devices must agree on the exact timing (Baud Rate) beforehand.

I2C (Inter-Integrated Circuit)

- **Wiring:** Requires 2 signal wires (SDA for data, SCL for clock) plus Ground. It is a synchronous connection.
- **Advantages:** It is very efficient for connecting many devices using only two wires because it uses an addressing system. It is ideal for sensors.
- **Disadvantages:** It is slower than SPI and requires more complex software handling (addressing overhead). It is generally not suitable for long cables due to capacitance issues.

SPI (Serial Peripheral Interface)

- **Wiring:** Requires 3 to 4 signal wires (MOSI, MISO, SCK, CS) plus Ground. It is a synchronous connection.
- **Advantages:** It offers very high data transfer rates and full-duplex communication (sending and receiving at the same time).
- **Disadvantages:** It requires many I/O pins. It lacks a built-in error checking mechanism and is designed strictly for very short distances (usually on the same circuit board).

2. Selection of Peer-to-Peer Connection

To establish communication between the separate Car Traffic Light and the Pedestrian Traffic Light, the most suitable connection is **UART**.

Justification:

1. **Signal Stability:** Traffic light models are often physically separated by a distance of a few meters. UART is much more reliable over these cable lengths than I2C or SPI, which are designed for short, on-board connections.
2. **Independence:** The two traffic lights run independent timing loops. UART is asynchronous, meaning the devices do not need to share a synchronized clock signal. They can operate independently and simply process messages when they arrive in the buffer.
3. **Simplicity:** The system only needs to exchange simple status flags (e.g., "Car Light is Red"). UART achieves this with the simplest wiring configuration (Cross-connecting TX to RX) without the need for complex slave addressing or pull-up resistors.

3. Generalization & Interoperability

Connecting your microcontroller to another colleague's system requires adapting the system to handle unknown hardware and software environments.

Do you need to upgrade your system?

Yes, upgrades are necessary for reliability:

- **Software Upgrade (Interrupts):** If the current code uses `delay()` functions, the system stops reading signals while waiting. To connect reliably to another colleague's system, the code must be upgraded to use interrupts. This ensures that incoming signals are captured immediately, regardless of what the main loop is doing.

Can you generalize your solution?

Yes, the solution can be generalized by creating a Communication Protocol.

- Instead of writing code that depends on specific internal logic, you define a standardized "language" of bytes. For example, agreeing that the byte 0x01 always means "Stop" and 0x02 always means "Go."
- By abstracting the communication into these standard bytes, your system becomes interoperable. It allows any other microcontroller to interact with your traffic light as long as they send the correct bytes, regardless of how their internal code is written.