Communication Protocols in Embedded Systems

Understanding Serial vs Parallel Communication

Communication between devices can happen in two fundamental ways, each with distinct characteristics and use cases.

Serial Communication

- **Definition**: One bit is transferred at a time over a single data line
- **Advantages**:
  - Requires fewer wires (typically 2-3 lines)
  - Smaller circuit footprint
  - Better for long-distance communication
  - Less susceptible to electromagnetic interference
- **Modern reality**: Today's serial protocols often outperform parallel due to advanced signaling techniques

Parallel Communication

- **Definition**: Multiple bits transferred simultaneously over multiple data lines
- **Historical context**: Initially parallel communication was faster than serial communication
- **Limitations**:
  - Parallel communication needs more wires and hence bigger circuit
  - Signal skew issues at high frequencies
  - More complex routing and higher cost

Note: While parallel was historically faster, modern high-speed serial protocols like USB 3.0, PCIe, and SATA have largely replaced parallel interfaces due to their superior performance and reliability.

Popular Serial Communication Protocols

Several serial communication protocols have been developed for different applications:

SUNBEAM INFOTECH

Protocol Categories by Application

General Purpose Protocols:

- **RS232** - Legacy but still widely used
- **USB** - Universal Serial Bus for computer peripherals
- **PS2** - Older keyboard/mouse interface

Embedded System Protocols:

- **I2C** (Inter-Integrated Circuit)
- **SPI** (Serial Peripheral Interface)
- **CAN** (Controller Area Network)
- **JTAG** - For debugging and programming

Classification by Distance

Short Distance Communication:

- **I2C**: Typically up to 1 meter, ideal for on-board communication
- **SPI**: Very short distances, usually within the same PCB

Long Distance Communication:

- **RS232**: Can work over several meters with proper cables

Noisy Environment Communication:

- **CAN**: Designed for automotive applications with high electromagnetic interference

Classification by Communication Direction

Simplex Communication:

- *One-way communication only*
- Example: Simple sensor data transmission

SUNBEAM INFOTECH

Half Duplex Communication:

- *Bidirectional but not simultaneous*
- **I2C**: Uses shared data line with addressing
- **CAN**: Bus-based system with collision detection

Full Duplex Communication:

- *Simultaneous bidirectional communication*
- **RS232**: Separate transmit and receive lines
- **SPI**: Independent MOSI and MISO lines

Classification by Network Topology

Peer to Peer:

- **RS232**: Direct one-to-one connection
- **PS2**: Point-to-point device connection

Bus Topology:

- **SPI**: Master-slave bus with chip select lines
- **I2C**: Multi-master/multi-slave bus with addressing
- **CAN**: Multi-node bus with arbitration
- **USB**: Tree topology with hub expansion

UART vs USART: Understanding the Difference

UART (Universal Asynchronous Receiver Transmitter)

- **Clock Generation**: Each device generates its own clock internally
- **Synchronization**: Both devices must be configured for the same **baud rate**
- **Limitation**: Clock drift can cause communication errors over time
- **Simplicity**: Easier to implement and configure

SUNBEAM INFOTECH

USART (Universal Synchronous Asynchronous Receiver Transmitter)

- **Enhanced Capability**: Supports both synchronous and asynchronous modes
- **Clock Sharing**: In synchronous mode, one device generates and shares clock with the other
- **Data Frame**: Single byte communication in a data frame
- **Performance**: Can work at higher speeds than UART due to shared clock
- **Advanced Features**: Supports complex protocols like:
  - **RS-485**: Multi-point differential signaling
  - **IrDA**: Infrared communication
  - **LIN**: Local Interconnect Network for automotive
  - **Smart Card**: ISO 7816 compliant communication
  - **ModBus**: Industrial communication protocol

RS-232: The Legacy Standard Still in Use

RS-232 remains one of the most important serial communication standards, especially in industrial and legacy applications.

Physical Characteristics

Communication Type

- **Full-duplex**: Simultaneous bidirectional communication
- **Peer-to-peer**: Direct device-to-device connection

Connection Requirements

Minimum Connection (3-wire):

- **Tx (Transmit)**: Data output line
- **Rx (Receive)**: Data input line
- **GND (Ground)**: Common reference voltage

Full Connection (with Handshaking):

SUNBEAM INFOTECH

- **CTS (Clear To Send)**: Hardware flow control from receiver
- **RTS (Request To Send)**: Hardware flow control from transmitter
- *These handshaking signals prevent data loss during transmission*

Connector Standards

- **Legacy**: DB-25 (25-pin connector) - largely obsolete
- **Standard**: DB-9 (9-pin connector) - most common today
  - **Half Serial Cable**: Only 3 essential wires connected (Rx, Tx, Ground)
  - **Full Serial Cable**: All 9 wires connected including handshaking signals

Voltage Levels

RS-232 uses **inverted logic** with specific voltage ranges:

- **Logic 0 (Space)**: +3V to +25V
- **Logic 1 (Mark)**: -3V to -25V
- **Encoding**: NRZ (Non-Return-to-Zero) - signal doesn't return to zero between bits

Voltage Conversion:

```
TTL Logic (0V/5V) ↔ MAX-232 IC ↔ RS-232 Levels (±3V to ±25V)
```

The MAX-232 IC is commonly used to convert between TTL and RS-232 voltage levels

Standard Baud Rates

Common rates include: **9600, 19200, 38400, 57600, 115200 bps** *Higher baud rates require better quality cables and shorter distances*

Logical Characteristics

Data Frame Structure

SUNBEAM INFOTECH

Every RS-232 transmission follows this frame format:

1. **Start Bit**: Always 0 (space) - signals beginning of data
2. **Data Bits**: 5 to 9 bits (typically 8 bits) - **LSB transmitted first**
3. **Parity Bit** (optional): Error detection mechanism
4. **Stop Bit(s)**: Always 1 (mark) - signals end of data frame

Parity Options

- **Even Parity**: Total number of 1-bits (including parity) is even
- **Odd Parity**: Total number of 1-bits (including parity) is odd
- **Sticky 1/0**: Parity bit always set to 1 or 0 regardless of data
- **No Parity**: Parity bit omitted to increase data throughput

Stop Bit Configuration

- **1 Stop Bit**: Sufficient for lower baud rates
- **2 Stop Bits**: Recommended for higher baud rates to ensure reliable frame detection

Error Detection and Handling

Common Error Conditions:

1. **Parity Error**:
    - Received parity doesn't match expected parity based on configuration
    - Indicates possible bit corruption during transmission
2. **Frame Error**:
    - Stop bit received as 0 instead of expected 1
    - Usually indicates timing synchronization issues
3. **Read Overrun**:

SUNBEAM INFOTECH

- Microcontroller fails to read received data before next byte arrives
- Previous data gets overwritten, causing data loss
- *Solution: Use hardware/software flow control or increase processing speed*

4. **Noise Error**:

- Detected through oversampling techniques (especially in STM32 microcontrollers)
- Indicates electrical noise interference on communication lines

SUNBEAM INFOTECH