What is SPI and how does it work?
SPI is a synchronous serial communication protocol that uses a master-slave architecture. It requires four lines: SCLK (clock), MOSI (Master Out Slave In), MISO (Master In Slave Out), and SS/CS (Slave Select/Chip Select). The master controls the clock and initiates all communications. Data is transmitted simultaneously in both directions on each clock cycle.

What is I2C and how does it work?
I2C is a synchronous serial communication protocol that uses only two wires: SDA (Serial Data) and SCL (Serial Clock). It supports multiple masters and slaves on the same bus. Communication starts with a START condition, followed by addressing, data transfer, and ends with a STOP condition. It includes acknowledgment bits for error detection.

What are the advantages and disadvantages of SPI?
Advantages: High speed, full-duplex communication, simple implementation, no start/stop bits needed, supports multiple slaves. Disadvantages: Requires more pins (4 wires minimum), no acknowledgment mechanism, no flow control, limited distance due to separate clock line

What are the advantages and disadvantages of I2C?
Advantages: Only two wires needed, supports multiple masters and slaves, built-in addressing, acknowledgment mechanism, standardized protocol. Disadvantages: Slower than SPI, more complex protocol, limited by bus capacitance, potential for bus conflicts with multiple masters.

How do you connect multiple slaves in SPI?
Multiple slaves can be connected using individual chip select lines for each slave, or by daisy-chaining slaves where MISO of one connects to MOSI of the next. The first method allows independent communication with each slave, while daisy-chaining treats all slaves as one large shift register.

How does I2C addressing work?
I2C uses 7-bit or 10-bit addressing. In 7-bit mode, the address is followed by a read/write bit (0 for write, 1 for read), making it 8 bits total. Each device on the bus must have a unique address. Some addresses are reserved for special functions.

What are SPI modes and how do they differ?
SPI has four modes based on Clock Polarity (CPOL) and Clock Phase (CPHA):
Mode 0: CPOL=0, CPHA=0 (clock idle low, data sampled on rising edge)
Mode 1: CPOL=0, CPHA=1 (clock idle low, data sampled on falling edge)
Mode 2: CPOL=1, CPHA=0 (clock idle high, data sampled on falling edge)
Mode 3: CPOL=1, CPHA=1 (clock idle high, data sampled on rising edge)

What are START and STOP conditions in I2C?
START condition: SDA line goes low while SCL is high, indicating the beginning of transmission. STOP condition: SDA line goes high while SCL is high, indicating the end of transmission. These conditions are only generated by the master and help synchronize communication on the bus.

What is the maximum speed of SPI?
SPI can operate at very high speeds, typically up to 50 MHz or higher, depending on the implementation and hardware capabilities. The speed is limited by factors like wire length, capacitive loading, and the capabilities of the master and slave devices.

What is clock stretching in I2C?
Clock stretching allows a slave device to hold the SCL line low to pause communication when it needs more time to process data. The master must wait until the slave releases SCL before continuing. This provides flow control and ensures data integrity.

Comparison Questions

When would you choose SPI over I2C?
Choose SPI when you need higher speed, full-duplex communication, simpler implementation, or when working with devices that don't require addressing (like sensors in a point-to-point configuration). SPI is better for applications requiring fast data transfer like displays or ADCs.

When would you choose I2C over SPI?
Choose I2C when you need to minimize pin count, want to connect multiple devices easily, require built-in addressing and acknowledgment, or when working with devices that naturally fit the I2C ecosystem (like EEPROMs, RTCs, or sensors in a sensor network).

How do you handle errors in SPI vs I2C?
SPI has no built-in error detection - error handling must be implemented at the application level using checksums or other methods. I2C has built-in acknowledgment bits that provide immediate feedback about successful data reception, making error detection easier.

What are the typical applications for SPI and I2C?
SPI applications: High-speed ADCs/DACs, flash memory, displays, sensors requiring fast sampling, shift registers. I2C applications: EEPROMs, RTCs, temperature sensors, accelerometers, device configuration, system monitoring.

How do you debug SPI and I2C communication issues?
For both protocols, use oscilloscopes or logic analyzers to examine signal integrity, timing, and protocol compliance. Check for proper pull-up resistors (I2C), correct wiring, proper power supply, and verify that both master and slave are configured for the same communication parameters (speed, mode for SPI, addressing for I2C). Software debugging includes checking initialization code, timing constraints, and proper handling of acknowledgments (I2C) or chip select signals (SPI).