To study and perform Serial Peripheral Interface with Arduino board 1. To perform SPI communication between Two Arduino boards 2. To perform one Master and multiple Slave communication between Arduino boards

**ChatGPT said:**

ChatGPT

Serial Peripheral Interface (SPI) is a synchronous communication protocol commonly used for short-distance communication between microcontrollers, sensors, and peripherals. Here’s a structured guide to study and perform SPI communication using Arduino:

**1. SPI Communication Between Two Arduino Boards**

**Objective:**

* Establish SPI communication between two Arduino boards.
* Set one Arduino as **Master** and the other as **Slave**.
* Send and receive data between the two boards.

**Connections:**

| **SPI Pin** | **Master (Arduino)** | **Slave (Arduino)** |
| --- | --- | --- |
| MISO | Pin 12 | Pin 12 |
| MOSI | Pin 11 | Pin 11 |
| SCK | Pin 13 | Pin 13 |
| SS | Pin 10 | Pin 10 |
| GND | GND | GND |

**Master Code:**

cpp

CopyEdit

#include <SPI.h>

void setup() {

SPI.begin();

pinMode(10, OUTPUT); // SS pin for Master

Serial.begin(9600);

}

void loop() {

digitalWrite(10, LOW); // Select Slave

byte dataToSend = 100; // Example data

byte receivedData = SPI.transfer(dataToSend);

digitalWrite(10, HIGH); // Deselect Slave

Serial.print("Received from Slave: ");

Serial.println(receivedData);

delay(1000);

}

**Slave Code:**

cpp

CopyEdit

#include <SPI.h>

volatile byte receivedData;

volatile byte dataToSend = 50; // Example data

void setup() {

pinMode(10, INPUT); // SS pin for Slave

SPI.begin();

SPCR |= \_BV(SPE); // Enable SPI in Slave mode

SPI.attachInterrupt(); // Enable SPI Interrupt

Serial.begin(9600);

}

ISR(SPI\_STC\_vect) {

receivedData = SPDR; // Read received data

SPDR = dataToSend; // Prepare response

Serial.print("Received from Master: ");

Serial.println(receivedData);

}

void loop() {

}

**2. One Master and Multiple Slaves Communication**

**Objective:**

* Connect **one Master** with **two Slaves** and communicate separately.
* The **Master selects each Slave using a separate SS pin**.

**Connections:**

| **SPI Pin** | **Master (Arduino)** | **Slave 1 (Arduino)** | **Slave 2 (Arduino)** |
| --- | --- | --- | --- |
| MISO | Pin 12 | Pin 12 | Pin 12 |
| MOSI | Pin 11 | Pin 11 | Pin 11 |
| SCK | Pin 13 | Pin 13 | Pin 13 |
| SS | Pin 10 (Slave 1), Pin 9 (Slave 2) | Pin 10 | Pin 10 |
| GND | GND | GND | GND |

**Master Code:**

cpp

CopyEdit

#include <SPI.h>

void setup() {

SPI.begin();

pinMode(10, OUTPUT); // SS for Slave 1

pinMode(9, OUTPUT); // SS for Slave 2

Serial.begin(9600);

}

void loop() {

// Communicating with Slave 1

digitalWrite(10, LOW);

byte received1 = SPI.transfer(101);

digitalWrite(10, HIGH);

Serial.print("Received from Slave 1: ");

Serial.println(received1);

delay(1000);

// Communicating with Slave 2

digitalWrite(9, LOW);

byte received2 = SPI.transfer(202);

digitalWrite(9, HIGH);

Serial.print("Received from Slave 2: ");

Serial.println(received2);

delay(1000);

}

**Slave 1 Code:**

cpp

CopyEdit

#include <SPI.h>

volatile byte receivedData;

volatile byte dataToSend = 55;

void setup() {

pinMode(10, INPUT);

SPI.begin();

SPCR |= \_BV(SPE);

SPI.attachInterrupt();

Serial.begin(9600);

}

ISR(SPI\_STC\_vect) {

receivedData = SPDR;

SPDR = dataToSend;

Serial.print("Received from Master: ");

Serial.println(receivedData);

}

void loop() {}

**Slave 2 Code:**

cpp

CopyEdit

#include <SPI.h>

volatile byte receivedData;

volatile byte dataToSend = 77;

void setup() {

pinMode(10, INPUT);

SPI.begin();

SPCR |= \_BV(SPE);

SPI.attachInterrupt();

Serial.begin(9600);

}

ISR(SPI\_STC\_vect) {

receivedData = SPDR;

SPDR = dataToSend;

Serial.print("Received from Master: ");

Serial.println(receivedData);

}

void loop() {}

**Conclusion:**

* **Single Slave:** SPI communication between two Arduino boards was successfully implemented.
* **Multiple Slaves:** A single Master was able to control two Slaves using separate SS pins.

Would you like additional features such as **sensor data transmission or display integration**? 🚀

4o

**You said:**

what is output in both que in both device

**ChatGPT said:**

ChatGPT

**Expected Output for Both Cases**

**1. SPI Communication Between Two Arduino Boards (One Master, One Slave)**

**Master Output (Serial Monitor)**

csharp

CopyEdit

Received from Slave: 50

Received from Slave: 50

Received from Slave: 50

...

* The **Master** sends 100 and receives 50 from the **Slave**.
* This repeats every second (delay(1000);).

**Slave Output (Serial Monitor)**

csharp

CopyEdit

Received from Master: 100

Received from Master: 100

Received from Master: 100

...

* The **Slave** receives 100 from the **Master**.
* It always replies with 50.

**2. One Master and Two Slaves (Multi-Slave SPI Communication)**

**Master Output (Serial Monitor)**

csharp

CopyEdit

Received from Slave 1: 55

Received from Slave 2: 77

Received from Slave 1: 55

Received from Slave 2: 77

...

* **Master sends 101 to Slave 1** and receives 55.
* **Master sends 202 to Slave 2** and receives 77.

**Slave 1 Output (Serial Monitor)**

csharp

CopyEdit

Received from Master: 101

Received from Master: 101

Received from Master: 101

...

* Slave 1 receives 101 and always responds with 55.

**Slave 2 Output (Serial Monitor)**

csharp

CopyEdit

Received from Master: 202

Received from Master: 202

Received from Master: 202

...

* Slave 2 receives 202 and always responds with 77.

**Summary of Communication:**

| **Device** | **Sent Data** | **Received Data** |
| --- | --- | --- |
| **Master → Slave 1** | 101 | 55 |
| **Master → Slave 2** | 202 | 77 |
| **Slave 1 → Master** | 55 | 101 |
| **Slave 2 → Master** | 77 | 202 |

Would you like any modifications, such as sending dynamic data? 🚀

4o

**You said:**

write code for esp8266 vs rsp8266 both que

2/2

**ChatGPT said:**

ChatGPT

**SPI Communication Between Two ESP8266 Boards**

Since the ESP8266 does not have dedicated hardware SPI support like Arduino, we use **software SPI** (via the SPI.h library).

**Pin Mapping for SPI on ESP8266:**

| **SPI Pin** | **ESP8266 (NodeMCU) Default** |
| --- | --- |
| MISO | D6 (GPIO12) |
| MOSI | D7 (GPIO13) |
| SCK | D5 (GPIO14) |
| SS (Slave Select) | D8 (GPIO15) |

**1. SPI Communication Between Two ESP8266 Boards (One Master, One Slave)**

**Connections:**

| **SPI Pin** | **Master (ESP8266)** | **Slave (ESP8266)** |
| --- | --- | --- |
| MISO | D6 (GPIO12) | D6 (GPIO12) |
| MOSI | D7 (GPIO13) | D7 (GPIO13) |
| SCK | D5 (GPIO14) | D5 (GPIO14) |
| SS | D8 (GPIO15) | D8 (GPIO15) |
| GND | GND | GND |

**Master Code (ESP8266 - NodeMCU)**

cpp

CopyEdit

#include <SPI.h>

#define SS\_PIN D8 // GPIO15 (Slave Select)

void setup() {

SPI.begin();

pinMode(SS\_PIN, OUTPUT);

Serial.begin(115200);

}

void loop() {

digitalWrite(SS\_PIN, LOW); // Select Slave

byte dataToSend = 100; // Example data

byte receivedData = SPI.transfer(dataToSend);

digitalWrite(SS\_PIN, HIGH); // Deselect Slave

Serial.print("Received from Slave: ");

Serial.println(receivedData);

delay(1000);

}

**Slave Code (ESP8266 - NodeMCU)**

cpp

CopyEdit

#include <SPI.h>

volatile byte receivedData;

volatile byte dataToSend = 50; // Example response data

void setup() {

pinMode(D8, INPUT); // SS pin for Slave

SPI.begin();

SPCR |= \_BV(SPE); // Enable SPI in Slave mode

SPI.attachInterrupt(); // Enable SPI Interrupt

Serial.begin(115200);

}

ISR(SPI\_STC\_vect) {

receivedData = SPDR; // Read received data

SPDR = dataToSend; // Prepare response

Serial.print("Received from Master: ");

Serial.println(receivedData);

}

void loop() {

}

**Expected Output**

* **Master (ESP8266) Serial Monitor Output**

csharp

CopyEdit

Received from Slave: 50

Received from Slave: 50

Received from Slave: 50

...

* **Slave (ESP8266) Serial Monitor Output**

csharp

CopyEdit

Received from Master: 100

Received from Master: 100

Received from Master: 100

...

**2. One Master and Multiple Slaves on ESP8266**

**Connections for Master and Two Slaves**

| **SPI Pin** | **Master (ESP8266)** | **Slave 1 (ESP8266)** | **Slave 2 (ESP8266)** |
| --- | --- | --- | --- |
| MISO | D6 (GPIO12) | D6 (GPIO12) | D6 (GPIO12) |
| MOSI | D7 (GPIO13) | D7 (GPIO13) | D7 (GPIO13) |
| SCK | D5 (GPIO14) | D5 (GPIO14) | D5 (GPIO14) |
| SS | D8 (GPIO15) (Slave 1) / D4 (GPIO2) (Slave 2) | D8 (GPIO15) | D8 (GPIO15) |
| GND | GND | GND | GND |

**Master Code for Multiple Slaves**

cpp

CopyEdit

#include <SPI.h>

#define SS1 D8 // GPIO15 for Slave 1

#define SS2 D4 // GPIO2 for Slave 2

void setup() {

SPI.begin();

pinMode(SS1, OUTPUT);

pinMode(SS2, OUTPUT);

Serial.begin(115200);

}

void loop() {

// Communicate with Slave 1

digitalWrite(SS1, LOW);

byte received1 = SPI.transfer(101);

digitalWrite(SS1, HIGH);

Serial.print("Received from Slave 1: ");

Serial.println(received1);

delay(1000);

// Communicate with Slave 2

digitalWrite(SS2, LOW);

byte received2 = SPI.transfer(202);

digitalWrite(SS2, HIGH);

Serial.print("Received from Slave 2: ");

Serial.println(received2);

delay(1000);

}

**Slave 1 Code**

cpp

CopyEdit

#include <SPI.h>

volatile byte receivedData;

volatile byte dataToSend = 55;

void setup() {

pinMode(D8, INPUT);

SPI.begin();

SPCR |= \_BV(SPE);

SPI.attachInterrupt();

Serial.begin(115200);

}

ISR(SPI\_STC\_vect) {

receivedData = SPDR;

SPDR = dataToSend;

Serial.print("Received from Master: ");

Serial.println(receivedData);

}

void loop() {}

**Slave 2 Code**

cpp

CopyEdit

#include <SPI.h>

volatile byte receivedData;

volatile byte dataToSend = 77;

void setup() {

pinMode(D8, INPUT);

SPI.begin();

SPCR |= \_BV(SPE);

SPI.attachInterrupt();

Serial.begin(115200);

}

ISR(SPI\_STC\_vect) {

receivedData = SPDR;

SPDR = dataToSend;

Serial.print("Received from Master: ");

Serial.println(receivedData);

}

void loop() {}

**Expected Output for Multi-Slave Communication**

* **Master (ESP8266) Serial Monitor Output**

csharp

CopyEdit

Received from Slave 1: 55

Received from Slave 2: 77

Received from Slave 1: 55

Received from Slave 2: 77

...

* **Slave 1 (ESP8266) Serial Monitor Output**

csharp

CopyEdit

Received from Master: 101

Received from Master: 101

Received from Master: 101

...

* **Slave 2 (ESP8266) Serial Monitor Output**

csharp

CopyEdit

Received from Master: 202

Received from Master: 202

Received from Master: 202

...

**Conclusion**

✅ **Single Slave Communication:** Master sends 100, receives 50.  
✅ **Multiple Slaves Communication:** Master sends 101 to Slave 1 (55 reply) and 202 to Slave 2 (77 reply).