EED308 Embedded System Hardware Spring 2021

Experiment 9 SPI Protocol

The objective of this experiment is to get familiar with establishing the serial communication between the STM32 microcontroller board and the Arduino Uno board using SPI Protocol. STM32 has **1 or more SPI bus** in it while Arduino Uno has **one SPI bus** so that it makes the communication possible between them. In this experiment we use STM32F303RE as Master and Arduino Uno as Salve to blink the LED which is connected with Arduino Uno.

Working of SPI:

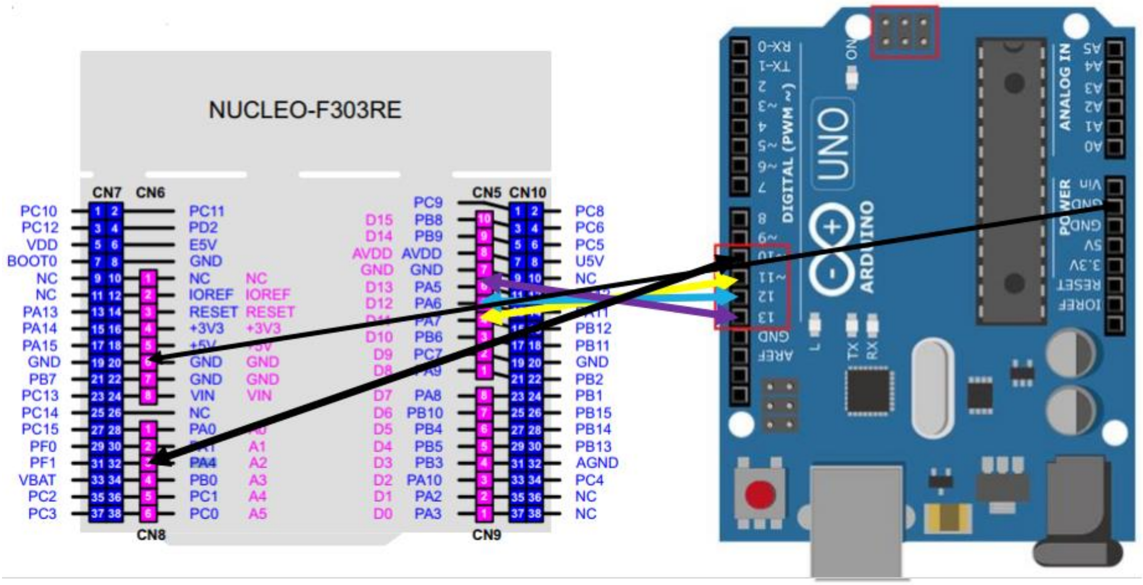
A SPI has a master/Slave communication by using four lines. A SPI can have only one master and can have multiple slaves. A master is usually a microcontroller and the slaves can be a microcontroller, sensors, ADC, DAC, LCD etc. To start communication between master and slave we need to set the Slave Select (SS) pin to LOW, so that it can communicate with the master. When it's high, it ignores the master. This allows you to have multiple SPI devices sharing the same MISO, MOSI, and CLK lines of master.

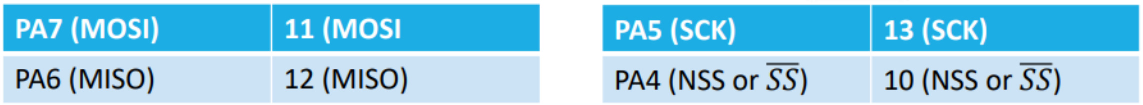
Components and Software applications required:

Components: Arduino Uno board, STM32F303RE board, LED, USB Cables and connecting wires.

Software applications: STM32CubeIDE and Arduino IDE.

Circuit Connection Diagram:

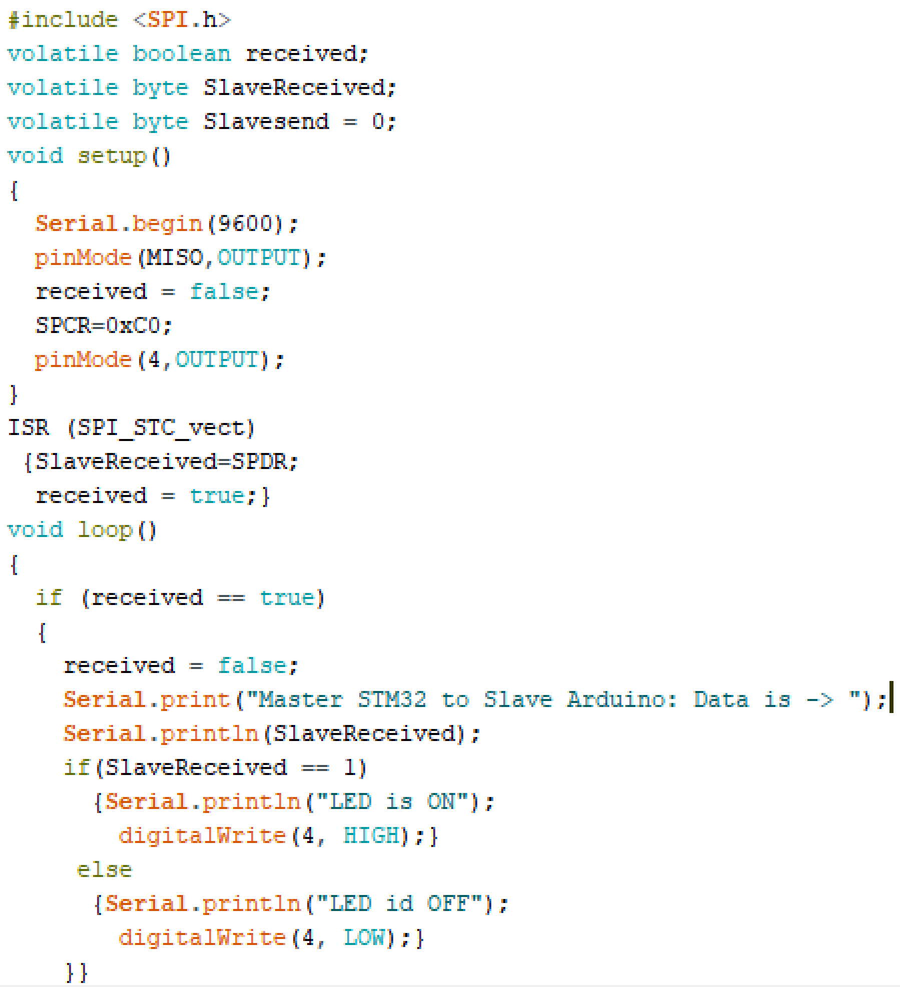




PA7 (MOSI)	11 (MOSI	PA5 (SCK)	13 (SCK)
PA6 (MISO)	12 (MISO)	PA4 (NSS or $\overline{SS}$)	10 (NSS or $\overline{SS}$)

Section: 01 Steps to program the Arduino Uno board

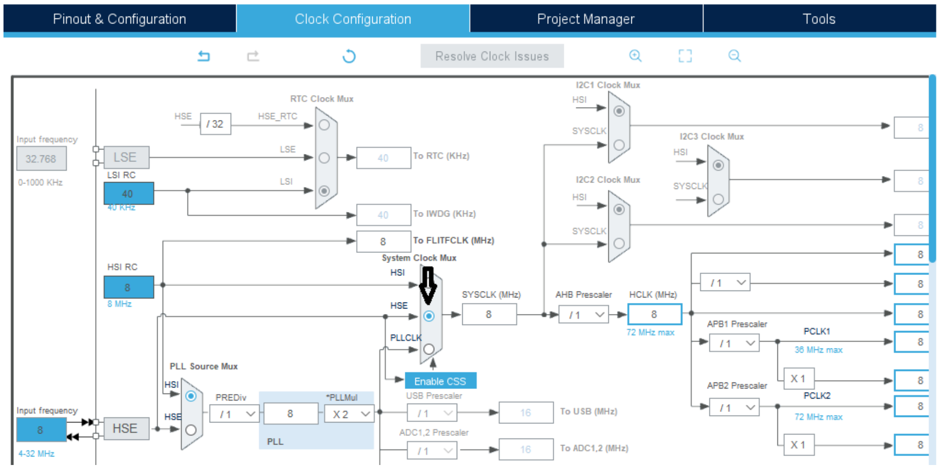
1. Open Arduino IDE and create the new file.
2. Select the serial port by clicking **Tools → Port → COMx (Arduino Uno)**
3. Write the program for blinking the LED as the program pasted below.
4. Save the file and compile (with no errors) then upload.



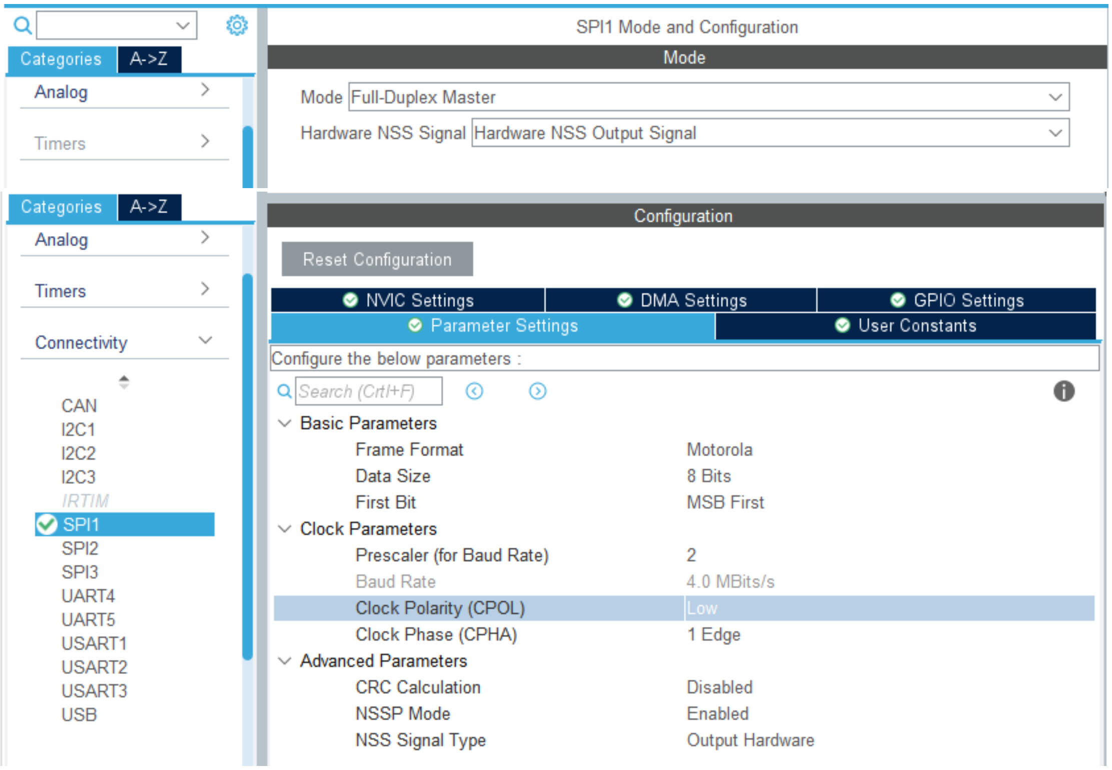
```
#include <SPI.h>
volatile boolean received;
volatile byte SlaveReceived;
volatile byte Slavesend = 0;
void setup()
{
  Serial.begin(9600);
  pinMode(MISO,OUTPUT);
  received = false;
  SPCR=0xC0;
  pinMode(4,OUTPUT);
}
ISR (SPI_STC_vect)
 {SlaveReceived=SPDR;
  received = true;}
void loop()
{
  if (received == true)
  {
    received = false;
    Serial.print("Master STM32 to Slave Arduino: Data is -> ");
    Serial.println(SlaveReceived);
    if(SlaveReceived == 1)
      {Serial.println("LED is ON");
        digitalWrite(4, HIGH);}
     else
      {Serial.println("LED id OFF");
        digitalWrite(4, LOW);}
    }}
```

Section: 02 Steps to program the STM32F303RE board

1. Open the STM32CubeIDE and create the new STM32 project.
2. Select the appropriate STM32 board by entering the part number and name the project.
3. In the **Pinout and Configuration** window,
   - Select **RCC** as High Speed Clock (HSE): Crystal/Ceramic Resonator.
4. In the **Clock Configuration** window,
   - Choose **HSE** from the multiplexer to provide the system clock (SYSCLK) as shown.



5. Select **SPI1** from pinout and configuration pane and do the following settings as depicted in below image.



SPI1 Mode and Configuration

Mode

Mode: Full-Duplex Master

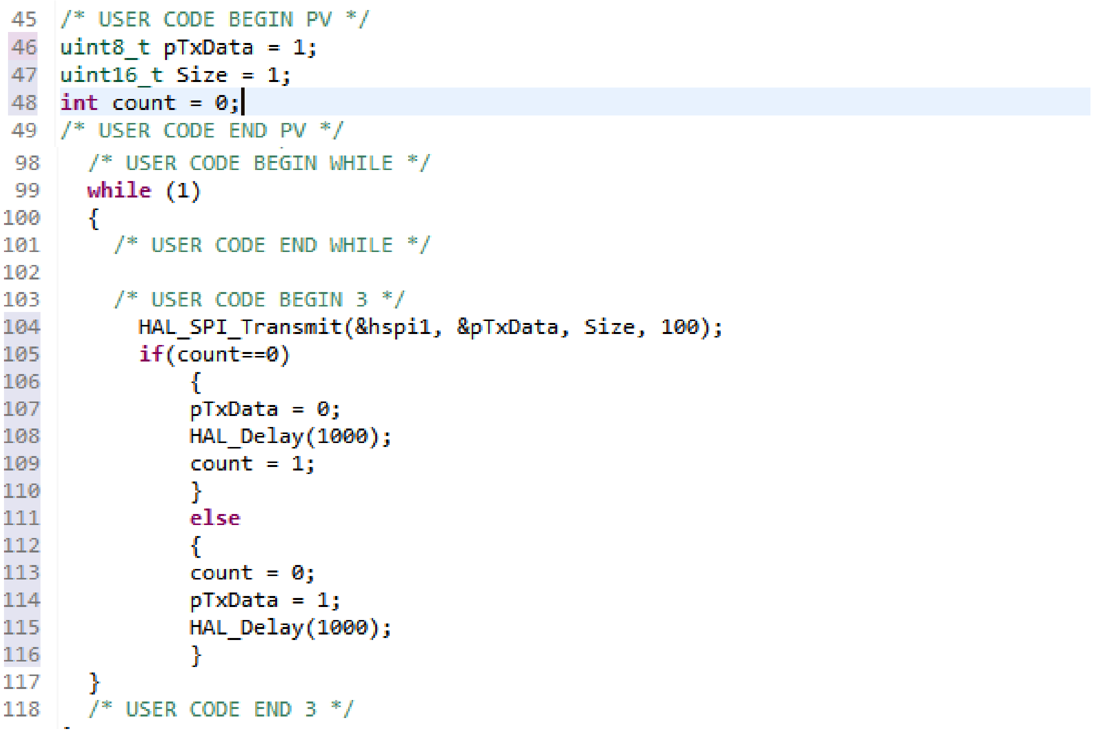
Hardware NSS Signal: Hardware NSS Output Signal

Configuration

Configure the below parameters :

Basic Parameters	
Frame Format	Motorola
Data Size	8 Bits
First Bit	MSB First
Clock Parameters	
Prescaler (for Baud Rate)	2
Baud Rate	4.0 MBits/s
Clock Polarity (CPOL)	Low
Clock Phase (CPHA)	1 Edge
Advanced Parameters	
CRC Calculation	Disabled
NSSP Mode	Enabled
NSS Signal Type	Output Hardware

6. After doing the above mentioned settings, generate the code and add the followings in the main.c file. (Under PV and BEGIN 3)

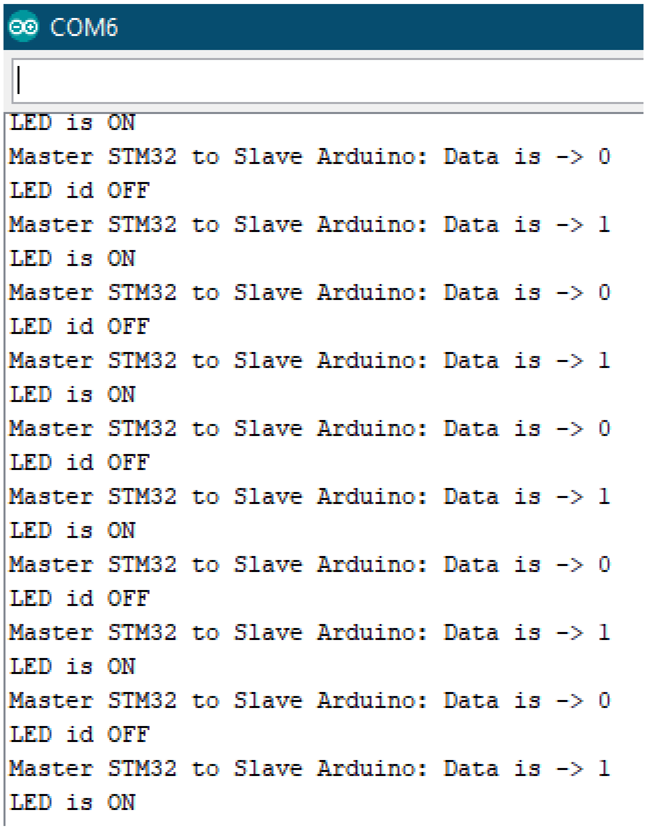


```
/* USER CODE BEGIN PV */
uint8_t pTxData = 1;
uint16_t Size = 1;
int count = 0;
/* USER CODE END PV */
```

```
  /* USER CODE BEGIN WHILE */
  while (1)
  {
    /* USER CODE END WHILE */

    /* USER CODE BEGIN 3 */
      HAL_SPI_Transmit(&hspi1, &pTxData, Size, 100);
      if(count==0)
          {
          pTxData = 0;
          HAL_Delay(1000);
          count = 1;
          }
          else
          {
          count = 0;
          pTxData = 1;
          HAL_Delay(1000);
          }
  }
  /* USER CODE END 3 */
```

7. Save and debug the program then click on Run. ▶ ▾
8. Open the Serial Montor window in the Arduino IDE and observe the LED blinking status.



```
LED is ON
Master STM32 to Slave Arduino: Data is -> 0
LED id OFF
Master STM32 to Slave Arduino: Data is -> 1
LED is ON
Master STM32 to Slave Arduino: Data is -> 0
LED id OFF
Master STM32 to Slave Arduino: Data is -> 1
LED is ON
Master STM32 to Slave Arduino: Data is -> 0
LED id OFF
Master STM32 to Slave Arduino: Data is -> 1
LED is ON
Master STM32 to Slave Arduino: Data is -> 0
LED id OFF
Master STM32 to Slave Arduino: Data is -> 1
LED is ON
Master STM32 to Slave Arduino: Data is -> 0
LED id OFF
Master STM32 to Slave Arduino: Data is -> 1
LED is ON
```

NOTE:

- There should have common ground (GND) connection between mater and slave devices.
- Clock polarity and clock phase of master and slave must be same.
- Frequency of the generated clock by master for SPI communication must be less than or equal to half of the slave's core clock frequency.