****

**AMERICAN INTERNATIONAL UNIVERSITY-BANGLADESH**

**Faculty of Engineering**

Lab Report

**Experiment # 06**

**Experiment Title:**  Communication between two Arduino Boards using SPI.

|  |  |  |  |
| --- | --- | --- | --- |
| **Date of Perform:** | 30th April 2025 | **Date of Submission:** | 7th May 2025 |
| **Course Title:** | Microprocessor and Embedded Systems Lab | | |
| **Course Code:** | EEE4103 | **Section:** | **R** |
| **Semester:** | Spring 2024-25 | **Degree Program:** | BSc in CSE |
| **Course Teacher:** | **Prof. Dr. Engr. Muhibul Haque Bhuyan** | | |

**Declaration and Statement of Authorship:**

1. I/we hold a copy of this Assignment/Case Study, which can be produced if the original is lost/damaged.
2. This Assignment/Case Study is my/our original work; no part has been copied from any other student’s work or any other source except where due acknowledgment is made.
3. No part of this Assignment/Case Study has been written for me/us by any other person except where such collaboration has been authorized.

by the concerned teacher and is acknowledged in the assignment.

1. I/we have not previously submitted or am submitting this work for any other course/unit.
2. This work may be reproduced, communicated, compared, and archived to detect plagiarism.
3. I/we permit a copy of my/our marked work to be retained by the Faculty Member for review by any internal/external examiners.
4. I/we understand that Plagiarism is the presentation of the work, idea, or creation of another person as though it is your own. It is a form of cheating and is a very serious academic offense that may lead to expulsion from the University. Plagiarized material can be drawn from, and presented in, written, graphic, and visual forms, including electronic data, and oral presentations. Plagiarism occurs when the origin of the source is not appropriately cited.
5. I/we also understand that enabling plagiarism is the act of assisting or allowing another person to plagiarize or copy my/our work.

*\* Student(s) must complete all details except the faculty use part.*

\*\* Please submit all assignments to your course teacher or the office of the concerned teacher.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Group # 05** | | | | |
|  |  |  |  |  |
| **Sl No** | **Name** | **ID** | **PROGRAM** | **SIGNATURE** |
| 1 | Shahriar Hossain | 22-48990-3 | BSc in CSE |  |
| 2 | Al Mubtasim | 22-49002-3 | BSc in CSE |  |
| 3 | Adiba Tanzila | 22-49012-3 | BSc in CSE |  |
| 4 | MD.Rakib Hasan | 22-49029-3 | BSc in CSE |  |
| 5 | Md.Imdadul Hasan(Ayon) | 22-49959-3 | BSc in CSE |  |

|  |  |  |
| --- | --- | --- |
| ***Faculty use only*** | | |
| FACULTY COMMENTS | **Marks Obtained** |  |
|  |
| **Total Marks** |  |

# Table of Contents

|  |  |
| --- | --- |
| Objectives | 3 |
| Equipment List | 3 |
| Circuit Diagram | 3 |
| Experimental Output Results (Color Photographs) | 4-5 |
| Simulation Output Results (Color Photographs) | 5-6 |
| Answers to the Questions in the Lab Manual | 7-11 |
| Discussion | 11 |
| References | 11 |

**Marking Rubrics (to be filled by Faculty):**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Level Category** | **Excellent**  **[5]** | **Proficient**  **[4]** | **Good**  **[3]** | **Acceptable**  **[2]** | **Unacceptable**  **[1]** | **No Response**  **[0]** |
| **Title and Objectives** | Able to clarify the understanding of the lab, no issues are missing and formatting is good. | Able to clarify the understanding of the lab experiment, no issues are missing but its formatting is not good. | Able to clarify the understanding of the lab experiment, but a few issues are wrong, and its formatting is bad. | Able to clarify the understanding of the lab experiment, but it lacks a few important issues of the experiment without maintaining the format. | Unable to clarify the understanding of the lab experiment. | No Response/ copied from others/ identical submissions with gross errors/image file printed |
| **Codes and Methods** | Able to explain the experimental codes and simulation methods using Proteus very well. | Able to explain the experimental codes and simulation methods using Proteus but is not formatted well. | Able to explain the experimental codes but simulation method using Proteus is not explained well. | Presents the experimental codes but didn’t explain simulation methods using Proteus clearly. | Presents the experimental codes but didn’t explain simulation methods using Proteus. |
| **Results** | Key results and images are there. Figures/Tables have all identifications and refer to them properly in the texts. | Key results and images are there. Figures/Tables have all identifications, such as the axis labels, numbers, and captions with a few minor errors; the texts refer them. | Key results and images are there. Figures/Tables lack a few identifications, such as the axis labels, numbers, and captions; the texts refer them. | Misses several key results and images. Figures/Tables lack identification, such as the axis labels, numbers, and captions; the texts don’t refer them. | Major results, such as experimental and simulation results’ images are not included. Figures and tables are poorly  constructed or not presented. |
| **Discussion and Conclusion** | Proper interpretation of results and summarizes the results to draw a conclusion, discusses its applications in real-life situations to connect with the report’s conclusion. | Proper interpretation of results and summarizes the results to draw a conclusion but didn’t discuss its applications in real-life situations to connect with the conclusion of the report. | Interpretation of results is presented. However, there is a disconnect between the results and discussion. | Misses the interpretation  of key results. There is little connection between the results and discussion. | Very poor interpretation of  the results. No connection  between results and discussions. |
| **Question and Answer** | Able to produce all questions’ answers correctly maintaining the lab report format. | Able to produce all questions’ answers but didn’t maintain the lab report format. | Able to produce all questions’ answers but wrong answers to a few questions. | Able to produce all questions’ answers but wrong/missing answers to multiple questions. | Unable to produce all questions’ answers and completely wrong answers. |
| **Comments** |  | | | | | **Total Marks (25)** |

**Objectives:**

The objectives of this experiment are to-

* Study the SPI protocol used in Arduino.
* Write assembly language programming code for SPI communication with Arduinos.
* Use SPI protocol for communication between two Arduinos.
* Build a circuit to control the master side LED by the push button at the slave side and vice versa using the SPI Serial communication protocol.
* Know the working principles of the SPI used in Arduino.

**Equipment List:**

1. Arduino IDE (2.0.1 or any recent version)

2. Arduino UNO (2)

3. LED (2)

4. Push Button (2)

5. Resistors 10 k, 2.2 k (2 + 2)

6. Breadboard

7. Connecting Wires

# Circuit Diagram:



**Fig : Two Arduino board’s pin connections for SPI communications (Schematic Diagram)**

**Experimental Output Results:**

A machine with wires and switches

AI-generated content may be incorrect.

**Fig. Hardware implementation of a LED light control using a switch from Master(“Controller”) and Slave(“Peripheral”)**

A screenshot of a computer

AI-generated content may be incorrect.

**Fig** Serial Monitor of Controller and Peripheral

**Simulation Output Results:**

A computer screen shot of a computer program

AI-generated content may be incorrect.

**Fig: Simulation of Both LED Off**

A computer screen shot of a computer program

AI-generated content may be incorrect.

**Fig: Simulation of Peripheral LED On**

A computer screen shot of a computer program

AI-generated content may be incorrect.

**Fig: Simulation of Controller LED On**

**Explaination**:

* Proteus was opened, and a new project was created.
* Required components such as Arduino Uno(both Controller and Peripheral), LEDs, resistors, buttons and Virtual terminals were added from the library.
* The components were connected accordingly.
* The arduino codes were compiled and HEX files for both Controller and Peripheral were created.
* The HEX file was loaded into the Arduino Unos in Proteus.
* The simulation was run, and the LED behavior for both Controller and Peripheral were observed using the switches.

**Answers to the Questions in the Lab Manual:**

**//Master/Controller Arduino Code:**

//SPI MASTER (ARDUINO)

//SPI COMMUNICATION BETWEEN TWO ARDUINO CIRCUIT DIGEST

#include<SPI.h> //Library for SPI

#define LED 7

#define ipbutton 2

int buttonvalue;

int x;

void setup (void){

Serial.begin(115200); //Starts Serial Communication at Baud Rate 115200

pinMode(ipbutton,INPUT); //Sets pin 2 as input

pinMode(LED,OUTPUT); //Sets pin 7 as Output

SPI.begin(); //Begins the SPI communication

SPI.setClockDivider(SPI\_CLOCK\_DIV8); //Sets clock for SPI communication at

// 8 (16/8 = 2 MHz)

digitalWrite(SS,HIGH); //Setting SS to HIGH do disconnect master from slave

}

void loop(void){

byte Mastersend, Mastereceive;

buttonvalue = digitalRead(ipbutton); //Reads the status of the pin 2

if(buttonvalue == HIGH) //Setting x for the slave based on input at pin 2

{

x = 1;

}

else

{

x = 0;

}

digitalWrite(SS, LOW); //Starts communication with Slave from the Master

Mastersend = x;

Mastereceive = SPI.transfer(Mastersend); //Sends the Mastersend value to

//the slave and also receives value from the slave

if(Mastereceive == 1) //To set the LED based on the value received from slave

{

digitalWrite(LED,HIGH); //Sets pin 7 HIGH

Serial.println("Master LED is ON");

}

else

{

digitalWrite(LED,LOW); //Sets pin 7 LOW

Serial.println("Master LED is OFF");

}

delay(1000);

}

**Explaination:**

**Arduino SPI Master Programming Explanation:**

1. First of all we need to include the SPI library for using SPI communication functions.

**#include<SPI.h>**

2. In void setup()

We Start Serial Communication at a Baud Rate of 1,15,200.

**Serial.begin(115200);**

Attach LED to pin 7 and Push button to pin 2 and set those pins as OUTPUT and INPUT respectively.

**pinMode(ipbutton,INPUT);**

**pinMode(LED,OUTPUT);**

Next, we begin the SPI communication.

**SPI.begin();**

Next, we set the Clockdivider for SPI communication. Here, we have set divider 8.

**SPI.setClockDivider(SPI\_CLOCK\_DIV8);**

Then we set the SS pin HIGH since we did not start any transfer from the Master to the slave Arduino.

**digitalWrite(SS,HIGH);**

3. In void loop():

We read the status of the push button pin connected to pin2 (Master Arduino) for sending those values to the slave.

**buttonvalue = digitalRead(ipbutton);**

Set Logic for setting x value (to be sent to the slave) depending upon the input from pin 2

if(buttonvalue == HIGH)

{

x = 1;

}

else

{

x = 0;

}

Before sending the value, we need to send LOW to the slave and select a value to begin the transfer to the slave from the Master.

**digitalWrite(SS, LOW);**

Then we send the push button value stored in *Mastersend* variable to the slave Arduino and also receive value from the slave that will be stored in *Mastereceive* variable.

**Mastereceive=SPI.transfer(Mastersend);**

After that, depending upon the *Mastereceive* value, we will turn the Master Arduino LED ON or OFF.

if(Mastereceive == 1)

{

digitalWrite(LED,HIGH); //Sets pin 7 HIGH

Serial.println("Master LED ON");

}

else

{

digitalWrite(LED,LOW); //Sets pin 7 LOW

Serial.println("Master LED OFF");

}

**//Slave/Peripheral Arduino Code:**

//SPI SLAVE (ARDUINO)

//SPI COMMUNICATION BETWEEN TWO ARDUINO

#include<SPI.h>

#define LEDpin 7

#define buttonpin 2

volatile boolean received;

volatile byte Slavereceived, Slavesend;

int buttonvalue;

int x;

void setup(){

Serial.begin(115200);

pinMode(buttonpin,INPUT); // Setting pin 2 as INPUT

pinMode(LEDpin,OUTPUT); // Setting pin 7 as OUTPUT

pinMode(MISO,OUTPUT); //Sets MISO as OUTPUT to send data to Master In

SPCR |= \_BV(SPE); //Turn on SPI in Slave Mode

received = false;

SPI.attachInterrupt(); //Interrupt ON is set for SPI communication

}

ISR(SPI\_STC\_vect) //Interrupt routine function

{

Slavereceived = SPDR; // Value received from Master stored in Slavereceived

received = true; //Sets received as True

}

void loop() {

if(received) //To set LED ON/OFF based on the value received from Master

{

if (Slavereceived == 1)

{

digitalWrite(LEDpin, HIGH); //Sets pin 7 as HIGH to turn on LED

Serial.println("Slave LED is ON");

}

else

{

digitalWrite(LEDpin,LOW); //Sets pin 7 as LOW to turn off LED

Serial.println("Slave LED is OFF");

}

buttonvalue = digitalRead(buttonpin); //Reads the status of the pin 2

if (buttonvalue == HIGH) //To set the value of x to send to Master

{

x = 1;

}

else

{

x=0;

}

Slavesend = x;

SPDR = Slavesend; //Sends the x value to the Master via SPDR

delay(1000);

}

}

**Explaination:**

**Arduino SPI Slave Programming Explanation:**

1. First of all we need to include the SPI library for using SPI communication functions.

**#include<SPI.h>**

2. In void setup()

We Start Serial Communication at Baud Rate of 1,15,200.

**Serial.begin(115200);**

Attach LED to pin 7 and Push button to pin2 and set those pins OUTPUT and INPUT respectively.

**pinMode(ipbutton,INPUT);**

**pinMode(LED,OUTPUT);**

We set MISO as OUTPUT (to send data to Master IN). So, data is sent via MISO of Slave Arduino.

**pinMode(MISO,OUTPUT);**

Now, turn on or enable the SPI in Slave Mode by using the SPI Control Register (bit 6 of SPCR).

**SPCR |= \_BV(SPE);**

Then turn ON interrupt for SPI communication. If data is received from the Master, the Interrupt Service Routine (ISR) is called and the received value is taken from SPDR (SPI Data Register)

**SPI.attachInterrupt();**

The value from the master is taken from SPDR and stored in *Slavereceived* variable. This takes place by following the Interrupt Routine function.

**ISR (SPI\_STC\_vect)**

**{**

Slavereceived = SPDR;

received = true;

**}**

3. Next in void loop(), we set the Slave Arduino LED to turn ON or OFF depending upon the Slavereceived value.

if (Slavereceived==1)

{

digitalWrite(LEDpin,HIGH); //Sets pin 7 as HIGH LED ON

Serial.println("Slave LED ON");

}

**else**

{

digitalWrite(LEDpin,LOW); //Sets pin 7 as LOW LED OFF

Serial.println("Slave LED OFF");

}

Next, we read the status of the Slave Arduino Push button and store the value in *Slavesend* to send the value to Master Arduino by giving value to SPDR register.

buttonvalue = digitalRead(buttonpin);

if (buttonvalue == HIGH)

{

x=1;

}

else

{

x=0;

}

Slavesend = x;

SPDR = Slavesend;

**Discussions**: This experiment successfully demonstrated SPI communication between two Arduino boards. The SPI protocol's synchronous and full-duplex nature was effectively utilized to enable bidirectional data exchange. The practical implementation validated the reliability of SPI in real-time device control. Connections like MISO, MOSI, SCK, and SS were correctly configured to establish communication. Although the LED did not light up as expected, possibly due to a wiring issue or a bug in the code,the overall SPI communication was verified through serial monitoring. Both simulation and hardware results (excluding the LED response) confirmed the system’s core functionality. Overall, the experiment provided valuable insights into SPI's practical applications and highlighted areas for further troubleshooting.

**Reference(s):**

[1] https://www.arduino.cc/.

[2] ATMega328 manual

[3] https://www.avrfreaks.net/forum/tut-c-newbies-guide-avr-timers

[4] http://maxembedded.com/2011/06/avr-timers-timer0/