Assignment I2C Communication

By July Trendafilov

St. nr.: 4674693

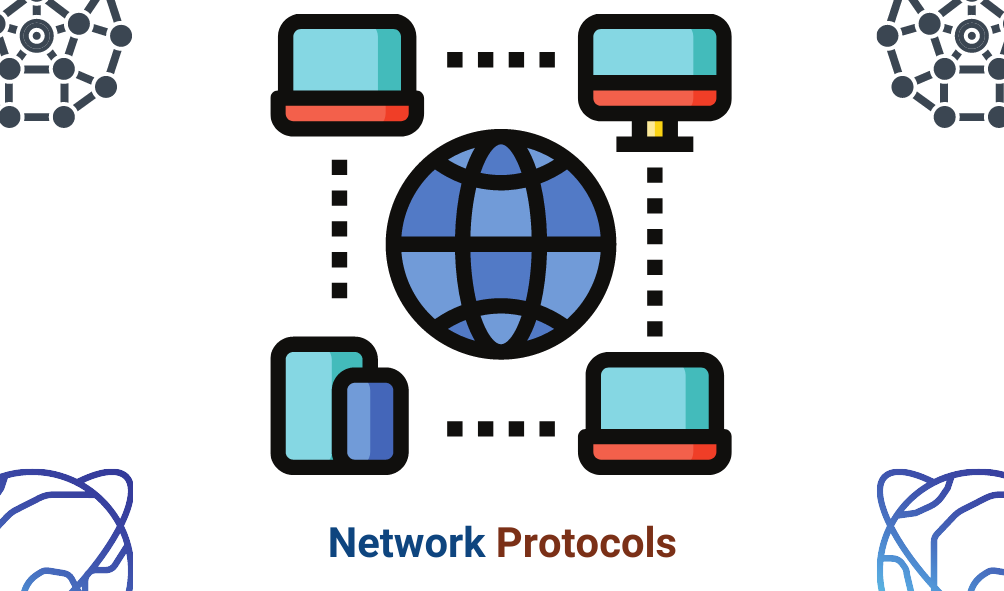


Table of Contents

[Version Table 2](#_Toc124934858)

[Introduction 3](#_Toc124934859)

[Introduction to I2C 3](#_Toc124934860)

[Design 3](#_Toc124934861)

[Application 3](#_Toc124934862)

[Protocol 4](#_Toc124934863)

[Table of messages 4](#_Toc124934864)

[Table of masters 4](#_Toc124934865)

[Diagram of communication 4](#_Toc124934866)

[Implementation 5](#_Toc124934867)

[Master1 5](#_Toc124934868)

[Master2 6](#_Toc124934869)

[Conclusion 9](#_Toc124934870)

[Bibliography 9](#_Toc124934871)

# Version Table

|  |  |
| --- | --- |
| Version 1 – 12.01.2023 | First release of the document |

# Introduction

In this assignment we must build a system using the I2C protocol which includes two Masters and two slaves which are respectively a Joystick and an OLED display. The point of this assignment is to achieve synchronization between the two masters.

# Introduction to I2C

The I2C is a protocol which allows a master to control multiple slaves (up to 110). Every slave in the network has an Address at which other slaves and the Master can send data. Because one Master controls multiple slaves there are common wires in the system:

* SCL – this line synchronizes the boards so that the communication is always working.
* SDA – this is the data line where all the data travels. On this line only one actor can send data to prevent data collisions.

# Design

The design of the system in terms of state machine is:

Diagram

Description automatically generated

# Application

The system consists of two masters that switch active state on button pressed. The first board is responsible for the number increment game where the user moves the joystick up or down to increment or decrement the value on the screen. The second board is responsible for the moving game where the user moves the icon using the joystick.

The common functionality is that when the button on the joystick is pressed, the boards switch active states or in other words the user switches the games.

# Protocol

To switch the active master on INIT one Master must start and the other must wait for permission to start. To do that they have a variable which is either 0 or 1 representing IDLE or ACTIVE. They can’t be both active at the same time. Whenever one is finished with the execution of its program, it goes into IDLE and send 1 to the other Master. Then the other master does its job and whets it is done, he goes into IDLE and send 1 to the other. This is the communication between the masters.

### Table of messages

|  |  |
| --- | --- |
| Message | Meaning |
| Byte 1 (00000001) | Transfer Active state |

### Table of masters

|  |  |
| --- | --- |
| Master | Address |
| Master 1 (Red board) | 0x09 |
| Master 2 (Arduino) | 0x10 |

### Diagram of communication

Diagram

Description automatically generated

Token

The two boards transfer ownership by sending each other 0x01. Whenever a Master sends 0x01 to an address, the initial state of the Master becomes inactive and wait for a value (0x01) to switch to active again. Whenever a Master sends the ownership of the system, they must go into IDLE state and listens for messages.

# Implementation

## Master1

Master1 has three functions:

1. receiveEvent() is a callback function so whenever we have an incoming data, it is called and processes the data. In this case it receives a 1 in order to switch to active state.

Text

Description automatically generated with medium confidence

1. setup() is the function where we setup the peripherals we will use. Here we connect the Joystick using its address and initialize the OLED and then clear its contents. Lastly we connect the callback function.

Text

Description automatically generated

1. The loop is the main function where we read if we increment or decrement the value and then print it to the OLED. If the button is pressed, we switch to IDLE state and send notification to the other master to go to active.

Text

Description automatically generated

## Master2

Master2 is similar to Master1.

1. It has the same callback function for synchronization of active and idle state between the two Masters. The only difference is that it sets the default coordinates on every startup so the game starts from the same coordinates.

A screenshot of a computer

Description automatically generated with medium confidence

1. The setup is the same because they use the same peripherals.

Text

Description automatically generated

1. The main loop is the application. We read the coordinates of the joystick to move the player around by changing the player’s coordinates. The player can’t go beyond the screen. When the button is pressed the game ends and the other game is started.

A screenshot of a computer

Description automatically generated with medium confidence

# Conclusion

The result is a system which has two games, one on each master and they control the peripherals. In this assignment we have learned to make a synchronization between two Masters in a I2C system.

# Bibliography

Dobbelsteen, J. (n.d.). *Github*. Retrieved from https://git.fhict.nl/technology/t-sem3-db/-/tree/master/communication/Toolbox

*Wikipedia*. (n.d.). Retrieved from https://www.wikipedia.org