Digital Systems Lab

Report: I2C Protocol Implementation with STM32F429

- Tushar Sharma (B21CS095)

1. Introduction

The purpose of this project is to demonstrate the implementation of the I2C (Inter-Integrated Circuit) communication protocol on an STM32F429 microcontroller. The project utilizes two serial expanders to control a total of eight LEDs, showcasing how I2C can be effectively employed for communication and control in embedded systems.

2. Hardware Setup

Components Used:

- STM32F429 board
- Two serial expanders
- Eight LEDs (four for each expander)
- Breadboard
- Jumper wires

Connections:

- Connect the STM32F429 board to the serial expanders via the I2C bus (SDA and SCL lines).
- Connect each serial expander to four LEDs on the breadboard.
- Ensure each serial expander has a unique I2C address.
- Use jumper wires to establish connections between components.

3. Software Setup

Tools Used:

- Keil uVision IDE

Steps:

I. Project Setup:

- Create a new project in Keil uVision IDE for the STM32F429 board.
- Add the necessary source files (`I2C_drivercode.h`, `I2C_drivercode.c`, `test.c`) to the project.

II. Configuration:

- Configure build settings and target device specifications in Keil uVision.

III. Building and Uploading:

- Build the project to generate the binary file.
- Upload the compiled code to the STM32F429 board for execution.

IV. Project Execution

- Initialization:
- Upon power-up, the STM32F429 initializes the I2C protocol.
- LEDs connected to the serial expanders are prepared for control.

- Control:
- The test program (`test.c`) sends specific commands to the serial expanders to control the LEDs.
- Addresses for the serial expanders are specified in the test program.

4. Code Explanation:

- **`header.h`**

This header file (`header.h`) contains declarations and macros necessary for implementing the I2C protocol. Here's a brief explanation of its contents:

- `I2C_CLOCK_FREQ`: Macro defining the clock frequency for the I2C communication.
- Function prototypes:
  - `void I2C_Init(void)`: Initializes the I2C peripheral.
  - `void I2C_Start(void)`: Generates a START condition on the I2C bus.
  - `void I2C_Stop(void)`: Generates a STOP condition on the I2C bus.
  - `void I2C_Write(uint8_t data)`: Writes data to the I2C bus.
  - `uint8_t I2C_Read_ACK(void)`: Reads data from the I2C bus with acknowledgment.
  - `uint8_t I2C_Read_NACK(void)`: Reads data from the I2C bus without acknowledgement.

- **`src.c`**

This source file (`src.c`) contains the implementation of the I2C protocol functionalities. Here's a breakdown of its functions:

- `I2C_Init()`: Initializes the I2C peripheral by configuring GPIO pins and setting up the I2C registers.
- `I2C_Start()`: Generates a START condition on the I2C bus to initiate communication.
- `I2C_Stop()`: Generates a STOP condition on the I2C bus to terminate communication.
- `I2C_Write(uint8_t data)`: Writes data to the I2C bus by sending the address and data byte.
- `I2C_Read_ACK()`: Reads data from the I2C bus with acknowledgment, indicating further data reception.
- `I2C_Read_NACK()`: Reads data from the I2C bus without acknowledgment, indicating the end of data transmission.

- **`test.c`**

This test program (`test.c`) demonstrates the functionality of the implemented I2C protocol. Here's what it does:

- `main()`: The entry point of the program.
- `I2C_Init()`: Initializes the I2C peripheral.
- Sends a START condition and writes data (0xAA) to a specified device address.
- Sends another START condition and reads data from the same device address.
- The received data is then stored (unused in this example).

The provided codes implement the I2C protocol for the STM32F429 microcontroller. The `header.h` header file contains function prototypes and macros, while the `src.c` source file implements the I2C protocol functionalities. The `test.c` test program demonstrates the usage of the implemented I2C protocol by initializing the I2C peripheral, writing data to a device, and reading data from the same device.

5. Results and Conclusion

The project successfully demonstrates the implementation of the I2C protocol on the STM32F429 microcontroller. Through the use of serial expanders, control signals are transmitted to LEDs, showcasing effective communication and control capabilities. This project highlights the versatility and applicability of the I2C protocol in embedded systems.

6. Reference:

[1] The I2C_resource pdf
[2] Lab Manual
[3] https://www.st.com/en/evaluation-tools/stm32-nucleo-boards.html
[4]crome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://www.ti.com/lit/an/sbaa565/sbaa565.pdf?ts=1714967100966&ref_url=https%253A%252F%252Fwww.google.com%252F#:~:text=I2C%20is%20a%20two%2Dwire,and%20receive%20commands%20and%20data.