I2C (Inter-Integrated Circuit) Digital Systems Documentation

Assignment

Digital Systems (EEP3020)

Presented By:

Shashank Shekhar Asthana (B21CS093)
Soham Parikh (B21CS074)

Under Supervision of

Prof. Binod Kumar

॥ त्वं ज्ञानमयो विज्ञानमयोऽसि ॥

DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY
JODHPUR - 342037

1. Overview of I2C Protocol

The Inter-Integrated Circuit (I2C) protocol is a synchronous, multi-master, multi-slave, packet-switched, single-ended, serial communication bus. It is primarily used for communication between integrated circuits on a single circuit board.

2. Frame Format

The I2C protocol uses a simple frame format consisting of a START condition, followed by an address frame, data frames, and a STOP condition. The address frame contains 7 or 10 bits indicating the destination device, while the data frames consist of 8 bits of data each. The START and STOP conditions are signaled by changes in the SDA line while the SCL line is high.

3. Code Structure and Functionality

The code is divided into separate header and source files for better readability and maintainability.

Header File (`I2C_DS_Assignment_B21CS093_B21CS074.h`)

- This file contains the necessary declarations and macros for the I2C implementation.
- It includes function prototypes for initialization, writing to, and reading from I2C devices.
- The macros define constants for the I2C clock frequency and any other necessary configurations.

Source File (`I2C_DS_Assignment_B21CS093_B21CS074`.c)

- This file contains the implementation of the I2C protocol functionalities.
- The `i2c_init` function initializes the I2C peripheral with the necessary configurations.
- The `i2c_start` and `i2c_stop` functions generate the START and STOP conditions, respectively.
- The `i2c_write` function sends data to an I2C device, while the `i2c_read` function reads data from an I2C device.

4. Comparison with Other Protocols

4.1 I2C vs. SPI

- Number of Lines: I2C uses two lines (SDA and SCL), while SPI uses four lines (SCLK, MOSI, MISO, and SS).
- Speed: SPI is generally faster, supporting speeds up to tens of MHz, while I2C typically operates up to 400 kHz (standard mode) or 3.4 MHz (high-speed mode).

- Complexity: I2C is more complex in terms of protocol overhead, while SPI is simpler.
- Addressing: I2C supports multiple devices on the same bus using addresses, while SPI uses separate select lines for each device.

4.2 I2C vs. UART

- Mode: I2C is synchronous, while UART is asynchronous.
- Number of Lines: I2C uses two lines, while UART uses two lines (TX and RX).
- Speed: UART can support a wide range of baud rates, while I2C has fixed speed modes.

4.3. I2C vs. CAN

- Use Case: CAN is primarily used in automotive applications, while I2C is used for general-purpose communication between integrated circuits.
- Number of Lines: CAN uses two lines (CANH and CANL), while I2C uses two lines (SDA and SCL).
- Speed: CAN supports higher speeds (up to 1 Mbps), while I2C is generally slower.

5. Application Areas

I2C is particularly advantageous in applications where multiple integrated circuits need to communicate on a single board, such as in embedded systems, sensor networks, and LCD displays.

I2C Protocol Implementation with STM32F429 board

This project demonstrates an I2C communication setup using an STM32F429 board and two serial expanders. Each expander controls four LEDs, which receive input from the serial data line and serial control line. The setup utilizes the addresses defined in the code to control the LEDs individually.

Project Structure

- `I2C_DS_Assignment_B21CS093_B21CS074.h` - Header file containing necessary declarations and macros.
- `I2C_DS_Assignment_B21CS093_B21CS074.c` - Source file containing the implementation of the I2C functionalities.

Hardware Setup

1. Components:

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

2. Connections:

- Connect the STM32F429 board to the serial expanders via the I2C bus (SDA and SCL lines).
- The serial expanders should be connected to the LEDs on the breadboard.
- Ensure that each expander has a unique I2C address.
- Use jumper wires to connect the components.

Software Setup

1. Keil uVision IDE:

- Open the Keil uVision IDE.
- Create a new project and select the STM32F429 board as the target device. Add the `I2C_DS_Assignment_B21CS093_B21CS074.h`, `I2C_DS_Assignment_B21CS093_B21CS074.c`, files to the project.
- Set up the build configurations as per the STM32F429 specifications.

2. Build and Upload:

- Build the project.
- Upload the compiled code to the STM32F429 board.

Running the Project

Upon powering up the board, the I2C protocol will initialize, and the LEDs connected to the serial expanders will be ready to respond to commands.

Conclusion

The I2C protocol is a versatile and efficient means of communication between integrated circuits. Its advantages include simplicity, low pin count, and multi-master capability. However, it may not be suitable for high-speed communication or long-distance transmission. This setup demonstrates the I2C communication protocol using the STM32F429 board and serial expanders to control LEDs. The implementation showcases how I2C can be used effectively in embedded systems for communication and control purposes.
