APPLICATION 12

I^2C COMMUNICATION

1. Purpose of the Application

In this lab you will learn how to communicate using the I^2C interface of the LPC2148 microcontroller. The I^2C is used to receive data from a temperature sensor and then displayed on a serial terminal.

2. Necessary Equipment

The equipment needed in this lab is the following:

- The MDK-ARM v4.00 or higher evaluation software from Keil (http://www.keil.com);
- A PC for writing the code;
- Embedded Artists LPC2148 Evaluation Board version 3.

3. Theoretical Considerations

The I^2C (Inter Integrated Circuit) is a two-wire serial communication system developed by Philips which allows multiple masters and multiple slaves connected via two wires: SCL (Serial Clock) and SDA (Serial Data). The SCL signal controls the data transfer between master and slave. The SDA signal is used to transmit or receive data. Both are open-drain bidirectional lines. The bit-rate is controlled by the SCL line. Common I²C bus speeds are the 100 kbit/s standard mode and the 10 kbit/s low-speed mode, but arbitrarily low clock frequencies are also allowed. Recent revisions of I²C can host more nodes and run at faster speeds (400 kbit/s Fast mode, 1 Mbit/s Fast mode plus or Fm+, and 3.4 Mbit/s High Speed mode). .

In an I^2C interface there is at least a single master and a single slave although I^2C can also support multi-master multi-slave configurations (Figure 1). The master generates the clock whereas the slave is driven by the clock.

There are four potential modes of operation for a device:

- Master transmit when the master device is sending data to a slave;
- Master receive when the master device is receiving data from a slave;
- Slave transmit when the slave device is sending data to the master;
- Slave receive when the slave device is receiving data from the master.

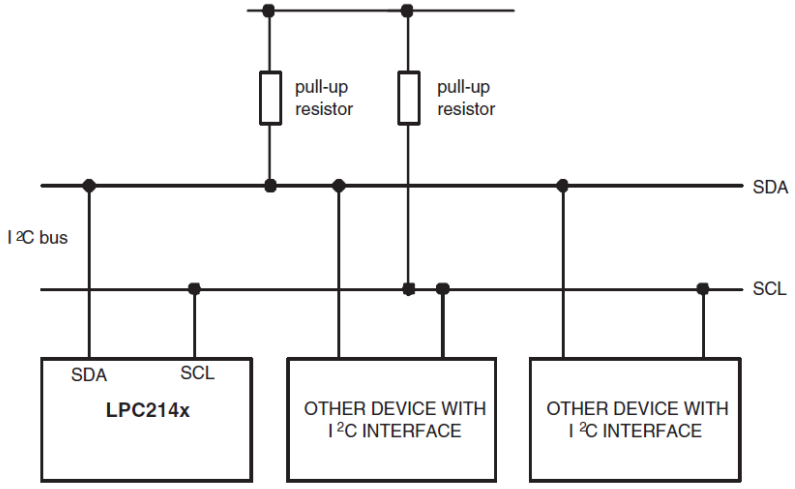

Figure 1- I2C Bus Configuration

To address one of the devices connected on the bus a 7-bit address space is used. The communication is initiated by the master device. When the master device wants to write or read from another device it will send a START bit followed by the 7-bit address of the slave device it wants to communicate with and then the following bit will specify if it wants to read (1) from the slave or write (0) to the slave. The slave with that address will answer with an acknowledgment bit (ACK). After this, the transfer of data can take place between the master and the slave. If the master is in receive mode, the slave will send data and the master will send ACK if data is received. If the master is in transmit mode the slave will receive data from the master and will respond with ACK if the data is received. For each byte (8 bits) of data an ACK is expected from the receiving device. The address and the data bytes are sent most significant bit first.

The master can end a transmission with an STOP bit, or take the bus by sending an START bit.

3.1 LPC2148 I^2C Interface

The LPC2148 I2C interface has the following features:

- Standard I2C compliant bus interfaces may be configured as Master, Slave, or Master/Slave;
- Arbitration is handled between simultaneously transmitting masters without corruption of serial data on the bus;
- Programmable clock allows adjustment of I2C transfer rates;
- Data transfer is bidirectional between masters and slaves;
- Serial clock synchronization allows devices with different bit rates to communicate via one serial bus;

- Serial clock synchronization is used as a handshake mechanism to suspend and resume serial transfer;
- I2C-bus can be used for test and diagnostic purposes.

The following registers are used to configure the LPC2148's I²C interface:

I2C Control Set register (I2CONSET: I2C0, I2C0CONSET - 0xE001 C000 and I2C1, I2C1CONSET - 0xE005 C000) The I2CONSET registers control setting of bits in the I2CON register that controls operation of the I2C interface. Writing a one to a bit of this register causes the corresponding bit in the I2C control register to be set. Writing a zero has no effect.

I2C Control Clear register (I2CONCLR: I2C0, I2C0CONCLR - 0xE001 C018 and I2C1, I2C1CONCLR - 0xE005 C018) The I2CONCLR registers control clearing of bits in the I2CON register that controls operation of the I2C interface. Writing a one to a bit of this register causes the corresponding bit in the I2C control register to be cleared. Writing a zero has no effect.

I2C Status register (I2STAT: I2C0, I2C0STAT - 0xE001 C004 and I2C1, I2C1STAT - 0xE005 C004) Each I2C Status register reflects the condition of the corresponding I2C interface. The I2C Status register is Read-Only.

I2C Data register (I2DAT: I2C0, I2C0DAT - 0xE001 C008 and I2C1, I2C1DAT - 0xE005 C008) This register contains the data to be transmitted or the data just received. The CPU can read and write to this register only while it is not in the process of shifting a byte, when the SI bit is set. Data in I2DAT remains stable as long as the SI bit is set. Data in I2DAT is always shifted from right to left: the first bit to be transmitted is the MSB (bit 7), and after a byte has been received, the first bit of received data is located at the MSB of I2DAT.

I2C Slave Address register (I2ADR: I2C0, I2C0ADR - 0xE001 C00C and I2C1, I2C1ADR - address 0xE005 C00C) These registers are readable and writable, and is only used when an I2C interface is set to slave mode. In master mode, this register has no effect. The LSB of I2ADR is the general call bit. When this bit is set, the general call address (0x00) is recognized.

I2C SCL HIGH duty cycle register (I2SCLH: I2C0, I2C0SCLH - 0xE001 C010 and I2C1, I2C1SCLH - 0xE0015 C010)

I2C SCL Low duty cycle register (I2SCLL: I2C0 - I2C0SCLL: 0xE001 C014; I2C1 - I2C1SCLL: 0xE0015 C014)

Software must set values for the registers I2SCLH and I2SCLL to select the appropriate data rate and duty cycle. I2SCLH defines the number of PCLK cycles for the SCL HIGH time; I2SCLL defines the number of PCLK cycles for the SCL low time. The frequency is determined by the following formula (PCLK is the frequency of the peripheral bus APB):

$$I^2C_{BitFreq} = \frac{PCLK}{I2SCLH + I2SCLL}$$

3.2 LM75A Temperature Sensor

The LM75A is a temperature-to-digital converter using an on-chip band gap temperature sensor and Sigma-delta A-to-D conversion technique. The device is also a thermal detector providing an overtemperature detection output. The LM75A contains a number of data registers: Configuration register (Conf) to store the device settings such as device operation mode, OS operation mode, OS polarity and OS fault queue; temperature register (Temp) to store the digital temp reading, and set-point registers (Tos and Thyst) to store programmable overtemperature shutdown and hysteresis limits, that can be communicated by a controller via the 2-wire serial I2C-bus interface. The device also includes an open-drain output (OS) which becomes active when the temperature exceeds the programmed limits. There are three selectable logic address pins so that eight devices can be connected on the same bus without address conflict.

The LM75A can be configured for different operation conditions. It can be set in normal mode to periodically monitor the ambient temperature, or in shutdown mode to minimize power consumption. The OS output operates in either of two selectable modes: OS comparator mode or OS interrupt mode. Its active state can be selected as either HIGH or LOW. The fault queue that defines the number of consecutive faults in order to activate the OS output is programmable as well as the set-point limits.

The temperature register always stores an 11-bit 2's complement data giving a temperature resolution of 0.125 °C. This high temperature resolution is particularly useful in applications of measuring precisely the thermal drift or runaway.

The device is powered-up in normal operation mode with the OS in comparator mode, temperature threshold of 80 °C and hysteresis of 75 °C, so that it can be used as a stand-alone thermostat with those pre-defined temperature set points.

For a full description of the sensor consult *LM75A Data Sheet*.

4. Experiments

4.1 Open the project

Download from the intranet lab12.rar and decompress it. Open lab12.uvproj. Complete the following exercises and answer the questions:

Exercise 1: Compile the project and download the HEX file. After the download has finished disable automatic ISP by removing the jumpers like in Fig 12.2.

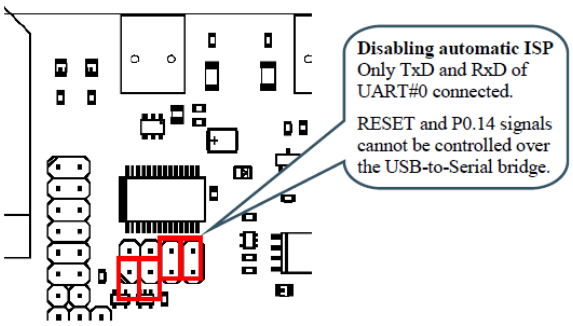

Figure 12.2 Disabling automatic ISP

Open a terminal program on the computer (eg. TeraTerm). Create a serial connection (Fig. 12.3) and then the go to *Setup->Serial…* menu and set up the connection with the parameters form Fig. 12.4

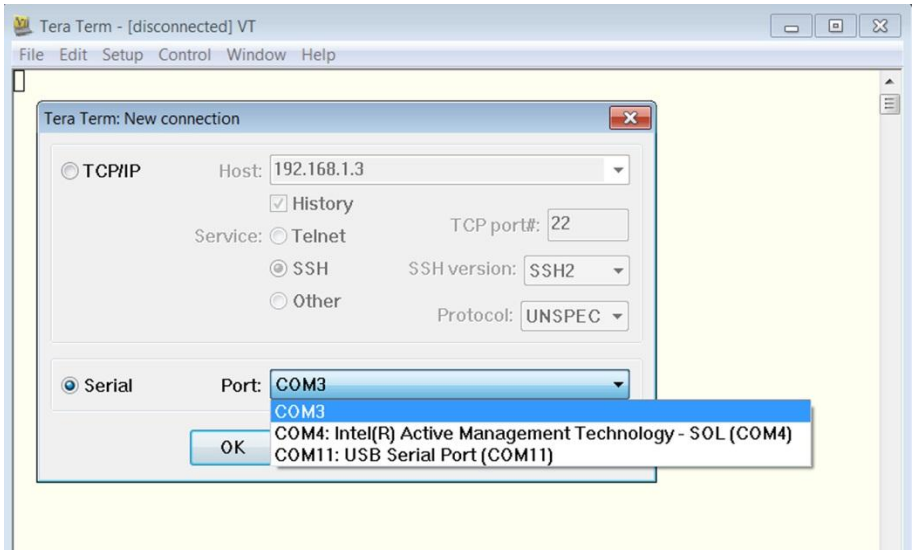

Fig. 12.3 Setting a new serial connection in TeraTerm

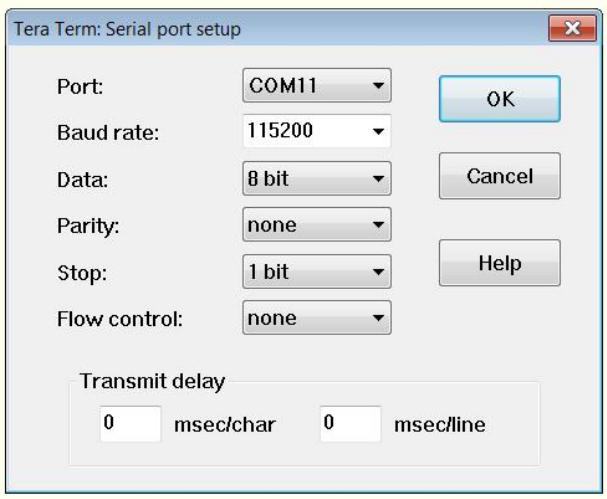

Fig. 12.4 Parameters for the serial connection

Press Reset on the board and see what happens.

Note: To download a new HEX file on the board, you have to connect the jumpers in their initial positions.

Exercise 2: Go to function i2c_init(). Search the registers in the *LPC2148 User Manual* and observe how the I^2C is set.

Exercise 3: Go to I2C_ISR() function. Explain what happens for each case of the switch instruction.

Note: The ***__irq*** *directive is used to specify that the function is a interrupt function.*

Exercise 3: The program presented only display correctly positive temperatures. Modify the program so it will display negative temperatures as well.