# **Embedded Systems Essentials with Arm: Get Practical with Hardware**

## Module 2

## SV1: Lab 2 - Further Serial: Part 1

In this lab, we will produce a fully functioning temperature measurement system, with the temperature displayed on an LCD and computer screen. It applies the “Inter-Integrated Circuit” or “I Two C” serial protocol, as well as continuing the use of the SPI and UART protocols introduced in the previous lab. The hardware builds directly on from the circuit we have built in the previous lab, so that will be our starting point.

First, let’s have a quick review of I Two C.

You will recall these figures from the knowledge material of this module. The first figure serves as a reminder that there are only two interconnecting wires on the I2C bus (apart from the ever-present ground and power interconnections). These are labelled SDA for data, and SCL for clock. Each wire must have a single pull-up resistor.

The I Two C data format, for a single data byte, is shown in the second figure. Data transmission, initiated by the master, starts with a Start condition. The first byte contains 7 bits of address, and a single read and write bit, followed by an acknowledge bit from the receiver. The second byte is the data byte. More data bytes can be sent. The message is terminated by a Stop condition.

The package of the DS1631 temperature sensor is shown here. Its full data sheet is found in the reference section, and is well worth looking at. The sensor comes as an 8-pin integrated circuit, with the ability to act as an I Two C slave. The device is made up of a semiconductor temperature sensor, an analog to digital converter, and a serial interface. It also contains a thermostat circuit, so the device can easily be used in simple temperature control applications.

The 7-bit address of the DS1631 will be inserted into the first byte of an I Two C communication, i.e. byte 1 of this figure. It is formed as shown here, i.e.by the 7-bit binary number 1 0 0 1 A2 A1 A0 . So, by wiring the address pins, pins 5-7, of the DS1631 to Ground or VDD, the user can select any of 8 possible addresses. As the least significant byte of the first message is “R W”, the slave address is often quoted with this least significant bit being set to 0. So, the DS1631 address can be set to be any even number from 10010000 (in hexadecimal: 0x90) to 10011110 (in hexadecimal: 0x9e).

This table shows the commands which are used to control the DS1631. The result of each temperature measurement is held in the 16-bit temperature register. The data is “left aligned” i.e. it occupies the full 8 bits of the more significant byte, and the three most significant bits of the less significant byte. The data format is “two’s complement”, which allows easy representation of a temperature below 0० C. However we are likely only to be dealing with positive temperatures in this application.

In this introductory experiment we use the DS1631 sensor in a simple way. There is a configuration register, but we make no adjustments to its default settings. Also we don’t make use of the facility to check if a conversion is complete, simply waiting for a fixed and possibly unnecessarily long period to allow completion.

Here we can see the circuit for the Temperature Sensing Unit. It’s easy to see that this builds directly on the final circuit of the previous lab, with the fairly straightforward DS1631 temperature sensor sub-circuit being added. Address pins 5, 6 and 7 of the DS1631 are connected to ground, which means that the temperature sensor address will be 1001000, or 0x90. According to I Two C requirements, two pull-up resistors are connected on the SDA and SCL. Values of 1k ohm are used for this prototype; for a final design this value could be optimised.

Connect up the circuit. If you’re building from scratch, you can omit the LCD at this stage, if you wish. The connection details of the Nucleo board are included here for convenience. A completed build is shown here.

Now, on to the actual program we will write for this lab. The program for the circuit needs to:

* Write the Start Convert command to the sensor;
* Wait for the conversion to complete, then write the Read Temperature command to the sensor;
* Read the 16-bit temperature data;
* Convert the binary temperature data into decimal; and
* Print the temperature to the PC via the UART (note that the LCD will be implemented later).

Here we can see the member functions of the Mbed I Two C API. These will be applied to make the data transfers, and the DS1631 commands to control the temperature sensor.

The I Two C API allows creation of an I Two C interface object that is connected to the specified pins. It enables the developer to set the frequency of the interface. It provides different read capabilities to read data from the I Two C slave or the bus and also some write functions to write data to the slave or bus.