Sensor Nodes Laboratory
Milestone Report 3
Data Encapsulation and Visualisation

Sebastian Thekkekera and Nitish Nagesh, Department of Electrical and Computer Engineering, Technical University of Munich, Arcisstr. 21, 80333 Munich, Germany

Introduction:

In the previous two reports, we gave an overview about the read-out circuit, filter design and interface between the analog front end and wireless communication board. In this report, we will elaborate the final prototype of the wireless sensor node and illustrate temperature data received remotely every two seconds

Hardware setup:

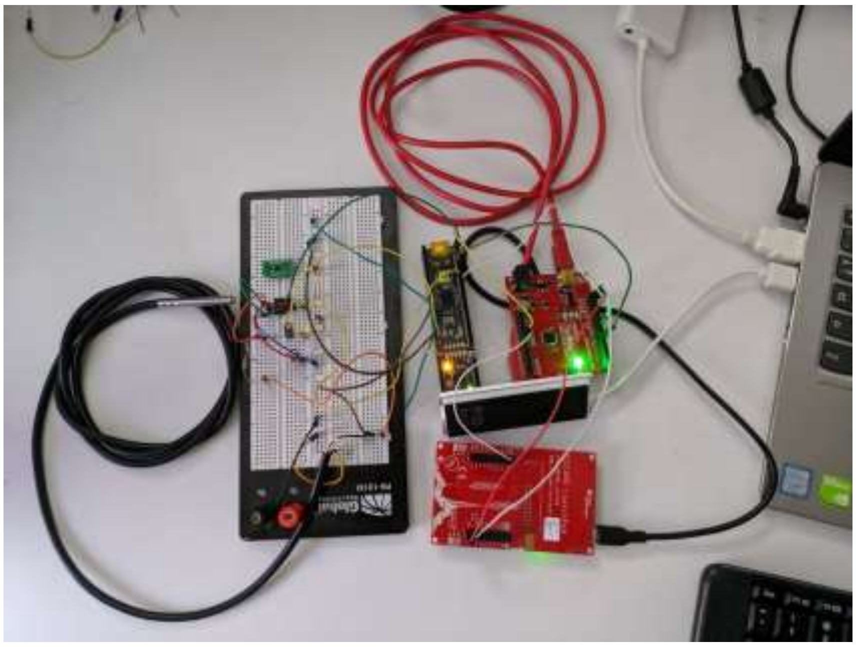

Fig. 1. Working prototype of wireless sensor node

We then tried implementing the UART communication protocol between the PSoC and launchpad. We were able to read temperature successfully on the terminal. Later, this was to be sent to a remote receiver and in this way, a wireless sensor node could be setup. But, we were not satisfied with this due to non-scalable nature of UART. UART uses one-on-one serial communication between two devices and does not allow addition of new devices to the set-up. This is a setback in practical applications which requires adaptation to new lines without much change in the hardware architecture.

To improve this design, we came up with a new idea. The main problem as highlighted before was that I2C communication was not successful between PSoC and CC1350 launchpad. However, the I2C communication between arduino and launchpad was working perfectly. Further, the communication between arduino and PSoC was not a problem. Therefore we decided to communicate between the

PSoC and arduino using UART and between arduino and launchpad via I2C. This implementation is shown in Fig. 1. The other aspects of the analog front-end were not changed.

The data was enqueued on the bus and the resistance was calculated. The corresponding temperature was calculated according to the equation [1]

$$R_T = R_0(1 + AT + BT^2)$$

for temperatures above 0 degree centigrade and similar equation for temperatures below 0 degree centigrade [1].

The resistance calculated in milli-ohm is a two byte data. This data is split into two components of one byte each and then transferred to the CC1350 Launchpad. Based on the calculated resistance value and coefficients from the above equation , a relationship between temperature and resistance is obtained. In the Launchpad, they are again combined to form a 16 bit value which is used to calculate the temperature.The corresponding values of resistance and temperature was compared with values in the look-up table [2]. It was found that there was only a slight deviation of both values when measurements were carried out at room temperature. One reason for the deviation can be attributed to interference from external noise. Another reason is the offset voltage between the inverting and non-inverting of the op-amp.

After the temperature was calculated, it was transmitted wirelessly over the server. This was possible via a specific address internet protocol (IP) address assigned to the launchpad. Fig. 1 shows the data received in a remote location. In this way, we were able to complete all the tasks assigned and implement a fully functional wireless sensor node.

Fallback strategy and scaling up:

While implementing this project, we faced considerable challenges with respect to hardware and software tools. It is of utmost importance that the sensor data is available at regular intervals. If the temperature is not read every two seconds an alert can be generated requesting a trigger from the terminal manually. The program can also be modified such that the launchpad and connected devices are sent into sleep mode after actuation by an undesirable input. When normal operation is to be run, the I2C communication, for instance can be woken up form sleep mode. The IDAC current can also be supplied intermittently based on request. This also enables power saving.

Power saving is especially crucial while using sensor nodes for practical applications such as smart lighting, smart meters, home automation etc. It helps increase longevity of the device under use thereby contributing to significant cost saving in corporations and institutions. The I2C communication protocol enables easy scaling-up of the prototype due to the availability of multi-master slave. Additional sensors such as humidity, ambient light, pressure etc. can be added to the existing prototype by using current multiplexer (MUX) and ADC MUX. The code has to be modified appropriately and each parameter can be observed simultaneously.

References:

[1] http://www.cypress.com/AN70698

[2] Kongsberg Maritime AS, "Platinum resistance temperature sensors Pt100 (Pt1000)"