University of British Columbia
Electrical and Computer Engineering
ELEC291/ELEC292

Module 3 – Data Logging using Python

Introduction

Embedded systems are often designed to perform simple or repetitive tasks while connected to larger computers. Examples of such systems are found in many of today's computers: the mouse, keyboard, memory sticks, hard drive controllers, etc. For this module you will build one of such devices: an embedded digital thermometer using the N76E003 microcontroller system. The digital thermometer will serially transmit the temperature to a personal computer using the serial port. You will program the personal computer using either Matlab or Python to receive the temperature and conveniently present it in real time using a strip chart plot.

There are many free python distributions available. One that has all the functionality to complete this laboratory module is WinPython version 3 available at:

https://winpython.github.io/

References

A51 user manual included with the latest version of CrossIDE.

N76E003 user manual.

Python reference manual. Available online.

Laboratory

1) **Testing the Serial Port of the N76E003 Microcontroller.** Available in the course web page you'll find the program 'hello.asm'. This program prints "Hello, world!" in PuTTY running in a personal computer throughout the serial port of the N76E003 microcontroller. Compile, load, and test this program using the N76E003 microcontroller and verify that you can receive the message through the serial port of a computer using PuTTY.

   PUTTY is a free Telnet/SSH/Serial terminal that can be downloaded from http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html. It is possible to launch PUTTY directly from CrossIDE by pressing <Control>+T. Configure PuTTY to 115200 baud, 8 bits, parity none, 1 stop bits, and Flow Control 'none'. Also make sure the 'Complete path of PuTTY.exe' field points to a valid location.[1]

2) **Analog to Digital Converter (ADC) in the N76E003.** The N76E003 microcontroller includes a multi-channel 12-bit ADC converter. In the course

[1] For macOS and Linux you can use CoolTerm: https://freeware.the-meiers.org/

web page you'll find an ADC test program ('ADC_test.asm') for the N76E003 microcontroller. This program reads the analog input from channel 7 of the ADC which is assigned to P1.1 and connected pin to 14. The program then converts the read value to voltage and displays the result using the LCD.

Compile and load 'ADC_test.asm' and verify that the program behaves as expected. Attach the variable end of a potentiometer to pin 14 of the N76E003; attach one of the fixed ends of the potentiometer to 5V and the other fixed end to reference ground (GND). If you measure pin 14 of the N76E003 with the lab multimeter, it should match the value displayed in the LCD attached to the N76E003.

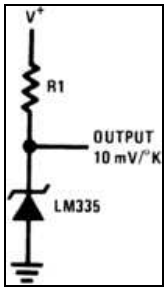
3) **Using Python to communicate with the N76E003 Microcontroller.** The Python script shown below opens the serial port in the host computer, constantly reads and prints a received value, and finally closes the serial port when CTRL+C is pressed in Python's command console. Attach an LM335 temperature sensor to ADC channel 7 of the N76E003. Modify the 'ADC_test.asm' program so it converts the acquired value to temperature and transmits it through the serial port every second. To convert the voltage acquired from the LM335 sensor to temperature, a library of 32-bit arithmetic functions ('math32.inc') and an example file ('mathtest.asm') are available in the course web page for you to use.

```
import time
import serial

# configure the serial port
ser = serial.Serial(
    port='COM1',
    baudrate=115200,
    parity=serial.PARITY_NONE,
    stopbits=serial.STOPBITS_TWO,
    bytesize=serial.EIGHTBITS
)
ser.isOpen()

while 1 :
    strin = ser.readline()
    print (strin)
```

The script above assumes you are using COM1. For other serial ports, adjust accordingly. Also, Python expects a new line escape sequence ('\n') for each received value from the microcontroller. Connect the LM335 as shown in the figure below. Make V^+=5V and R1=2kΩ. To observe different temperature readings, you can **carefully** heat up the LM335 using the solder iron.



4) **Temperature strip-chart using Python.** The script 'stripchart_sinewave.py' shows how to implement strip-charts in Python. A strip-chart can be used to plot the temperature transmitted from the N76E003 microcontroller to Python in real time. Modify the provided script so it plots the data received from the serial port.

   Demo the temperature strip-chart (in ºC) to you lab TA. Once again, you can use the solder iron to **carefully** heat the LM335 up! Don't forget to add extra functionality and/or features for bonus marks! Please upload to canvas:

   a) Python code.

   b) Assembly code.

   c) ONE picture of the microcontroller system.

   d) ONE Screen capture of temperature strip chart.

   You can put all the files in a ZIP file and upload the ZIP file instead. These files are not only needed to verify your work, but for program accreditation as well.