Lab 4: Multithreaded Programming and Image Processing **Due Date**: See the course schedule web page.

Objectives:

- Learn how to design multithreaded programs for embedded multicore systems
- Understand the principles of synchronization and mutual exclusion
- Learn POSIX thread libraries.
- Learn HTTP protocol and libcurl library
- Understand image processing concepts
- Implement image processing functions

Description:

You should by now have an embedded system with a variety of sensor devices such as light intensity, temperature, and camera. In this lab, you are tasked to send these data to a web server. You will need to design a multithreaded software program that communicates with both the sensor and a web server.

Your program should meet the following requirements:

1. Use at least three threads to perform the following tasks respectively

a. ***User Command Interface:*** this thread allows for local user commands such as resetting the PIC sensor, checking status (PING command: if you don't get an ACK report it as an "Error" status if you get an ACK report it as an "Alive" status), setting up interrupts and reading the PIC ADC values. It will also print the current ambient temperature from TMP102 sensor and the user will be able to set an new temperature threshold as a camera trigger (such as Lab3). This thread also facilitates debugging the program;

b. ***Sensor Control Center:*** this thread is responsible for sending commands to the sensors (LDR, TMP102 and Camera) and obtaining sensor responses. This thread also needs to check the sensor's status (if there is changes, i.e.: new image was taken, ambient light intensity changed dramatically, temperature sensor passed the threshold again) periodically and report its status to a web server (provided by the instructor);

c. ***Client-Server Communication:*** this thread communicates with the provided web server using HTTP protocol and *libcurl* library. The web based communication protocol is defined in Section 2.

d. Using OpenCV library capture an image and send it to the web server.

2. The sensor application (more specifically the communication thread) reports sensor status and data using HTTP POST method.

Your sensor application will use the following URL to supply status and data:

```
http://servername:portnumber/update?
id=var_xxxx&password=var_xxxx&name=var_xxxx&data=var_xxxx&status=var_xxxx&tim
estamp=var_xxxx&filename=var_xxxx
```

The sensor application must provide the following information as part of the URL request

id	a unique numerical identifier. Use your group number (example: if you are group 3 use "03" as ID #)
password	A unique password to authenticate the sensor application
name	The project group number or name (example: "Group03" or "Great_team")
status	the current status of the PIC sensor
data	an integer value of the ADC value (Built-in ADC of PIC16F18856)
timestamp	local time and date
filename	Name of the image file

Deliverables

A zipped file containing:

1. Source code with comments
2. Reports
3. Sensor Application Data flow diagram (.pdf or image file)

References

1. Posix Thread programming. Available at:

https://computing.llnl.gov/tutorials/pthreads/

2. libcurl APIs. Available at:

http://curl.haxx.se/libcurl/

EXTRA Credit (+ 1 Point):

For those who will give the option to the user from the User Command Interface thread

1. to change the TMP102 sampling frequency (showing the available options)
2. to set a temperature threshold in the temperature sensor (So, they will not have to set another temperature threshold in the Sensor Control Center thread). The sensor will trigger the camera with a direct TMP102 Register value