MEMORANDUM

TO: Dr. Christopher Peters
FROM: Yelnur Abilakim
DATE: 28 Oct 2021
SUBJECT: ECE-303 Lab 5: Graphical User Interfaces

Summary

In this experiment, the same circuit schematic is used as in the previous one. The duty cycle is being incremented from 1% to 100% in 1% increments. For each iteration, points are taken and plotted on the graph to visualize the system behavior. The project is now extended by adding a graphical user interface (GUI), while performing serial communication between the Arduino and MATLAB.

Introduction

In this experiment, the LED and photocell circuits from previous lab are kept intact. Unlike the previous experiment, the PWM approach can be replaced with analogWrite(). The goal of this lab is to implement automatic measuring and data acquisition, establish communication from the Arduino to MATLAB through the serial port, and display the results in MATLAB using graphical interface. The GUI has the following characteristics:

- A start button to start the process
- A text area to display the current duty cycle
- Plots of the following:
    - Photocell current versus duty cycle
    - Photocell voltage versus duty cycle
    - Photocell resistance versus duty cycle

Methods

Experimental apparatus used in this lab:

- Breadboard.
- Arduino MEGA 2560 connected as follows:
    - To laptop via USB cable.
    - To outlet via power cable.
- One 1kΩ resistor.
- One 10kΩ resistor.
- One photocell.
- Four jump wires.

The LED used in this experiment is a red LED in series with a 1kΩ resistor (either one can be connected to ground). On a different circuit, the photocell will be connected to the 5V pin, and in series with a 10kΩ resistor (resistor to ground). The circuit used for the lab is shown in Figure 1.

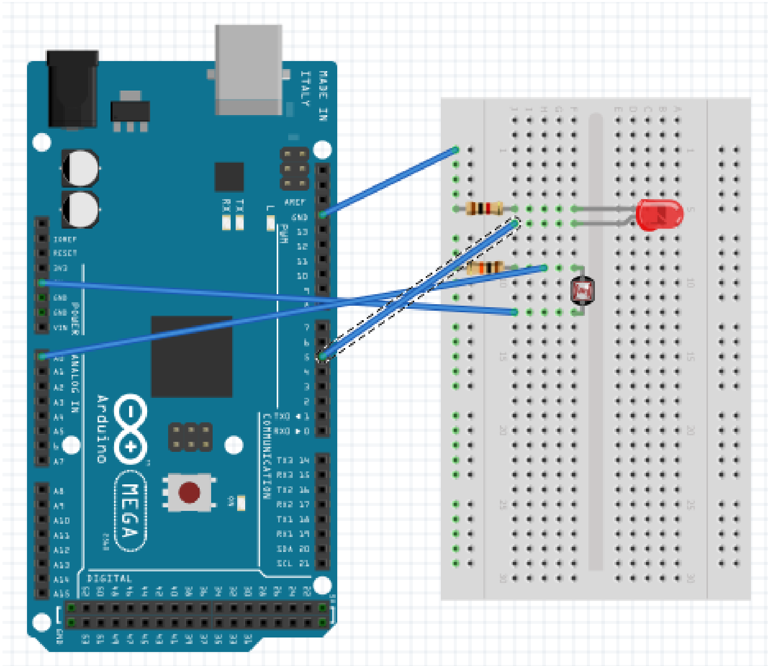

Figure 1. Circuit setup

Note that the LED should be placed near the photocell head, and photocell wiring needs to be bent such that the photocell head is in direct line with the LED. This minimizes energy losses, resulting in a more accurate measurement. In addition, the measurements should be performed in a dark environment so that external light sources do not interfere.

Results

The physical circuit used for the experiment, built according to Figure 1, is displayed in Figure 2.

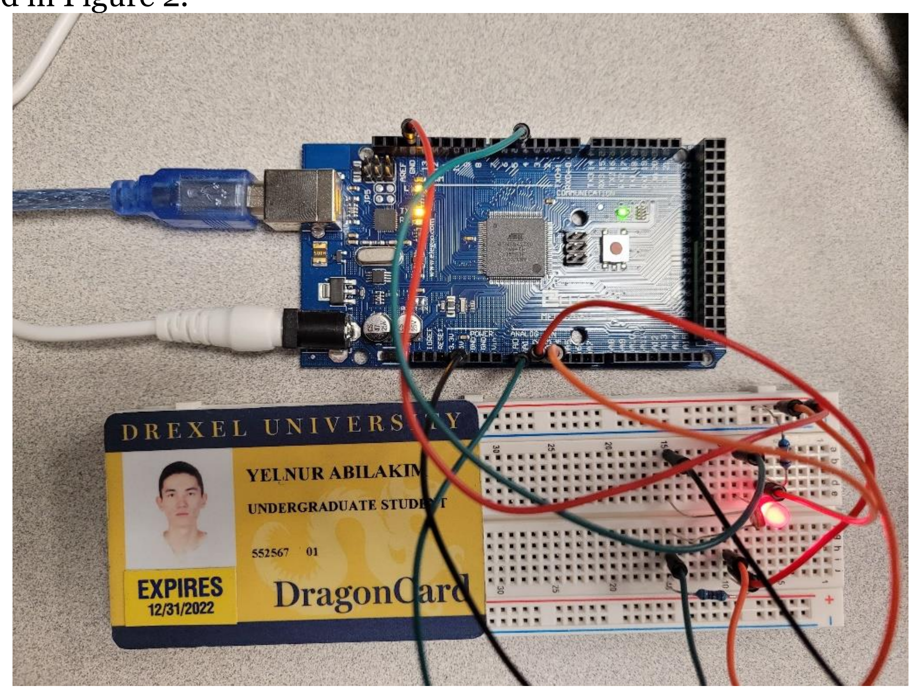

Figure 2. Physical circuit

Figure 3 demonstrates the plots created using the Serial Monitor outputs through the GUI in MATLAB window.

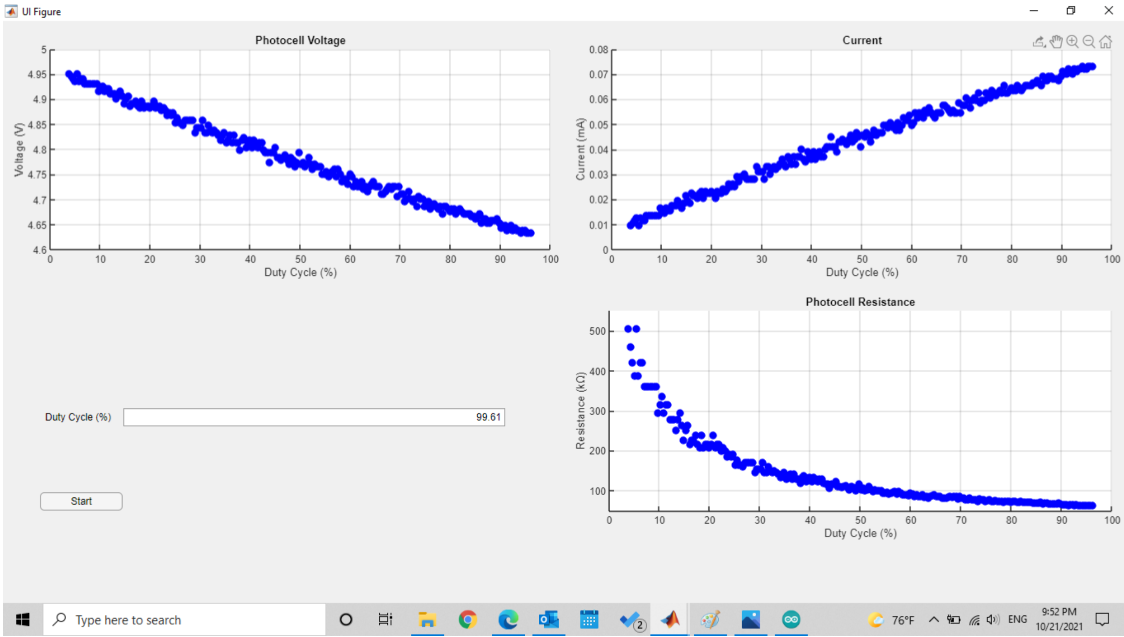

Figure 3. Plots

The Arduino and MATLAB codes used for this experiment are appended at the end.

Conclusions/Recommendations

The experiment was successful. Automatic measuring and data acquisition were implemented, and communication from the Arduino to MATLAB through the serial port was established, thus achieving the goal of the lab. The Serial Monitor has displayed all required plots as expected.

Appendices

Arduino code is shown in Figure 4.

```
const int pinLED = 5; // timer 3
const int pinPhoto = A0; // photocell at analog pin 0
const int pinRes = A1;
const int pinDuty = A2;
int val1 = 0;
int val2 = 0;
int val3 = 0;
unsigned int val = 0;
unsigned int counter = 0;

void setup(){
  Serial.begin(9600);
  pinMode(pinLED, OUTPUT);
}

void loop(){
    val = Serial.parseInt();
    analogWrite(pinLED, counter);
    delay(500);
    val1 = analogRead(pinPhoto);
    Serial.println(val1);
    val2 = analogRead(pinRes);
    Serial.println(val2);
    val3 = analogRead(pinDuty);
    Serial.println(val3);
    counter += 1;
}
```

Figure 6. Arduino code

The MATLAB script can be found in Figures 5-9.

```
classdef GUI < matlab.apps.AppBase

    % Properties that correspond to app components
    properties (Access = public)
        UIFigure                 matlab.ui.Figure
        UIAxes                   matlab.ui.control.UIAxes
        UIAxes_2                 matlab.ui.control.UIAxes
        UIAxes_3                 matlab.ui.control.UIAxes
        DutyCycleEditFieldLabel  matlab.ui.control.Label
        DutyCycleEditField       matlab.ui.control.NumericEditField
        StartButton              matlab.ui.control.Button
                                 matlab.ui.control.UIAxes
    end

    % Callbacks that handle component events
    methods (Access = private)

        % Button pushed function: StartButton
        function StartButtonPushed(app, event)
            arduino=serialport("COM5",9600,"Timeout",15);
            pause(1)
            num_points=255;
            DC=zeros(1,num_points);
            V_res=zeros(1,num_points);
            V_pc=zeros(1,num_points);
            I=zeros(1,num_points);
            R=zeros(1,num_points);

            for K=0:(num_points-1)
                DC(K+1)=K/255*100; % Duty Cycle
                flush(arduino)
                write(arduino,2,'string');
                pause(0.5)
```

Figure 5. MATLAB script (part 1)

```
                pause(0.5)
                a=read(arduino,4,'string');
                flush(arduino)
                V_res(K+1)=str2double(a)/1023*5;
                V_pc(K+1)=5-V_res(K+1);
                I(K+1)=V_res(K+1)/5000*1000; % Current (mA)
                R(K+1)=V_pc(K+1)/I(K+1); % Resistance (kOhm)
                app.DutyCycleEditField.Value=DC(K+1);
                plot(app.UIAxes,DC(1:(K+1)),V_pc(1:(K+1)),'bo','MarkerFaceColor','b')
                plot(app.UIAxes_2,DC(1:(K+1)),I(1:(K+1)),'bo','MarkerFaceColor','b')
                plot(app.UIAxes_3,DC(1:(K+1)),R(1:(K+1)),'bo','MarkerFaceColor','b')
            end
            delete(arduino)
        end
    end

    % Component initialization
    methods (Access = private)

        % Create UIFigure and components
        function createComponents(app)

            % Create UIFigure and hide until all components are created
            app.UIFigure = uifigure('Visible', 'off');
            app.UIFigure.Position = [100 100 640 480];
            app.UIFigure.Name = 'UI Figure';

            % Create UIAxes
            app.UIAxes = uiaxes(app.UIFigure);
            title(app.UIAxes, 'Photocell Voltage')
            xlabel(app.UIAxes, 'Duty Cycle (%)')
            ylabel(app.UIAxes, 'Voltage (V)')
            app.UIAxes.XGrid = 'on';
```

Figure 6. MATLAB script (part 2)

```
            app.UIAxes.YGrid = 'on';
            app.UIAxes.Position = [12 282 300 185];

            % Create UIAxes_2
            app.UIAxes_2 = uiaxes(app.UIFigure);
            title(app.UIAxes_2, 'Current')
            xlabel(app.UIAxes_2, 'Duty Cycle (%)')
            ylabel(app.UIAxes_2, 'Current (mA)')
            app.UIAxes_2.XGrid = 'on';
            app.UIAxes_2.YGrid = 'on';
            app.UIAxes_2.Position = [330 282 300 185];

            % Create UIAxes_3
            app.UIAxes_3 = uiaxes(app.UIFigure);
            title(app.UIAxes_3, 'Photocell Resistance')
            xlabel(app.UIAxes_3, 'Duty Cycle (%)')
            ylabel(app.UIAxes_3, 'Resistance (kΩ)')
            app.UIAxes_3.XGrid = 'on';
            app.UIAxes_3.YGrid = 'on';
            app.UIAxes_3.Position = [330 78 300 185];

            % Create DutyCycleEditFieldLabel
            app.DutyCycleEditFieldLabel = uilabel(app.UIFigure);
            app.DutyCycleEditFieldLabel.HorizontalAlignment = 'right';
            app.DutyCycleEditFieldLabel.Position = [46 159 86 22];
            app.DutyCycleEditFieldLabel.Text = 'Duty Cycle (%)';

            % Create DutyCycleEditField
            app.DutyCycleEditField = uieditfield(app.UIFigure, 'numeric');
            app.DutyCycleEditField.Position = [147 159 100 22];

            % Create StartButton
            app.StartButton = uibutton(app.UIFigure, 'push');
```

Figure 7. MATLAB script (part 3)

```
            app.StartButton.ButtonPushedFcn = createCallbackFcn(app, @StartButtonPush
            app.StartButton.Position = [46 57 100 22];
            app.StartButton.Text = 'Start';

            % Create
            app. = uiaxes(app.UIFigure);
            title(app., 'Title')
            xlabel(app., 'X')
            ylabel(app., 'Y')
            app..Position = [72 233 300 185];

            % Show the figure after all components are created
            app.UIFigure.Visible = 'on';
        end
    end

    % App creation and deletion
    methods (Access = public)

        % Construct app
        function app = GUI

            % Create UIFigure and components
            createComponents(app)

            % Register the app with App Designer
            registerApp(app, app.UIFigure)

            if nargout == 0
                clear app
            end
        end

```

Figure 8. MATLAB script (part 4)

```

        % Code that executes before app deletion
        function delete(app)

            % Delete UIFigure when app is deleted
            delete(app.UIFigure)
        end
    end
end
```

Figure 9. MATLAB script (part 5)