ECE 303 Lab 6

Drexel University

To: Dr. Christopher Peters

From: Zheren Gu

cc: Amirhosein Chahe

Date: 5/17/2023

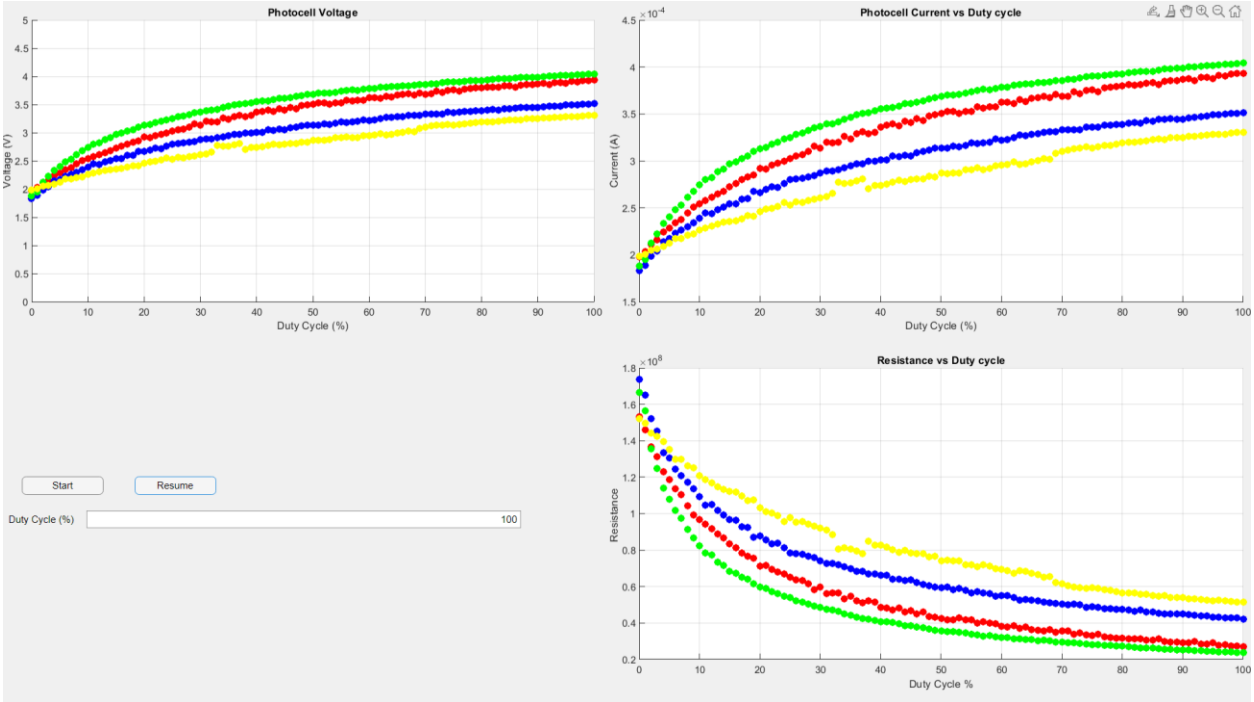
Re: Graphical User Interfaces

Purpose

The purpose of this lab was to use MATLAB to streamline the data measurement from last week, implementing a GUI instead of running MATLAB code by itself.

Discussion

The graphs measured in this lab are identical to the graphs from last week and the only difference is the GUI being used instead of the command line in MATLAB. Below is the GUI and the graphs that were obtained.



Recommendation

I don’t have any additions to make since the test and data collection of this lab was identical to the previous lab.

```
#include <Arduino.h>

#include "DigitalPin.h"


DigitalPin::DigitalPin(int pin)
```

```
{
  pinMode(pin, OUTPUT);
  _pin = pin;
}

void DigitalPin::set_ICR(){
   if(_pin == 6){
          TCCR4A = (1<<WGM41) | (1<<COM4A1);
          TCCR4B = (1<<WGM43) | (1<<CS40);
          ICR4 = 255;
          OCR4A = 50;
          TCNT4 = 0;
    }

}

void DigitalPin::set_duty_cycle(int num){
   if(_pin == 6){
          OCR4A = (num * 255) / 100;

    }

}
#ifndef DigitalPin_h
```

```
#define DigitalPin_h


#include <Arduino.h>


class DigitalPin

{

public:

  DigitalPin(int pin);


  void set_ICR();

  void set_duty_cycle(int val);

};

#endif
```

```
#include <Arduino.h>
#include <DigitalPin.h>

int photo = A0;
int val1 = 0;
unsigned int val = 0;
unsigned int count = 0;
DigitalPin LED(6);


void setup() {
  LED.set_ICR();
  Serial.begin(9600);

}
```

```
void loop() {
if (Serial.available() > 0){


  val=Serial.parseInt();
  LED.set_duty_cycle(count);
  delay(500);
  val1 = analogRead(photo);
  Serial.println(val1);
  count += 1;
  if (count == 101){
    count = 0;
  }
}
}
```

```
                    write(arduino, 2, 'string');
                    pause(0.5);
                    a = read(arduino, 4, 'string');
                    flush(arduino);
                    y(led, K+1) = (str2double(a)/1023)*5; % calculations


                    current(led, K+1) = y(led, K+1) / photoResistor;
                    resistance(led, K+1) = (photoResistor * (5 - (y(led, K+1))) ./
current(led, K+1));
                    disp([led, K, y(led, K+1)]);
                    if (led == 1) % Plotting
                      % Red LED
                      plot(app.UIAxes, K, y(led, K+1) , 'ro', 'MarkerFaceColor',
'red')
                      plot(app.UIAxes_2, K, current(led, K+1), 'ro',
'MarkerFaceColor', 'red')
                      plot(app.UIAxes_3, K, resistance(led, K+1), 'ro',
'MarkerFaceColor', 'red')
                      drawnow limitrate;
                    end
                    if (led == 2) % Plotting
                      % Blue LED
                      plot(app.UIAxes, K, y(led, K+1) , 'bo', 'MarkerFaceColor',
'blue')
                      plot(app.UIAxes_2, K, current(led, K+1), 'bo',
'MarkerFaceColor', 'blue')
                      plot(app.UIAxes_3, K, resistance(led, K+1), 'bo',
'MarkerFaceColor', 'blue')
                      drawnow limitrate;
                    end
                    if (led == 3) % Plotting
                      % Green LED
                      plot(app.UIAxes, K, y(led, K+1) , 'go', 'MarkerFaceColor',
'green')
                      plot(app.UIAxes_2, K, current(led, K+1), 'go',
'MarkerFaceColor', 'green')
                      plot(app.UIAxes_3, K, resistance(led, K+1), 'go',
'MarkerFaceColor', 'green')
                      drawnow limitrate;
                    end
```

```
                    if (led == 4) % Plotting
                      % Yellow LED
                      plot(app.UIAxes, K, y(led, K+1) , 'yo', 'MarkerFaceColor',
'yellow')
                      plot(app.UIAxes_2, K, current(led, K+1), 'yo',
'MarkerFaceColor', 'yellow')
                      plot(app.UIAxes_3, K, resistance(led, K+1), 'yo',
'MarkerFaceColor', 'yellow')
drawnow limitrate;
end
pause(0.01);
app.DutyCycleEditField.Value = K;
end


disp('Press any key to continue to the next LED...');
uiwait(app.UIFigure);
close;
end
hold(app.UIAxes, 'off');
hold(app.UIAxes_2, 'off');
hold(app.UIAxes_3, 'off');


delete(arduino);
clear arduino;




end


% Callback function
function EndButtonPushed(app, event)




end


% Value changed function: DutyCycleEditField
function DutyCycleEditFieldValueChanged(app, event)
value = app.DutyCycleEditField.Value;
app.DutyCycleEditField.Value = value;
end
```

```
% Button pushed function: ResumeButton
function ResumeButtonPushed(app, event)
uiresume(app.UIFigure);
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 = 'MATLAB App';


% Create UIAxes
app.UIAxes = uiaxes(app.UIFigure);
title(app.UIAxes, 'Photocell Voltage')
xlabel(app.UIAxes, 'Duty Cycle (%)')
ylabel(app.UIAxes, 'Voltage (V)')
zlabel(app.UIAxes, 'Z')
app.UIAxes.XLim = [0 100];
app.UIAxes.YLim = [0 5];
app.UIAxes.XGrid = 'on';
app.UIAxes.YGrid = 'on';
app.UIAxes.Position = [1 268 307 200];


% Create UIAxes_2
app.UIAxes_2 = uiaxes(app.UIFigure);
title(app.UIAxes_2, 'Photocell Current vs Duty cycle')
xlabel(app.UIAxes_2, 'Duty Cycle (%)')
ylabel(app.UIAxes_2, 'Current (A)')
zlabel(app.UIAxes_2, 'Z')
app.UIAxes_2.XLim = [0 100];
app.UIAxes_2.XGrid = 'on';
app.UIAxes_2.YGrid = 'on';
app.UIAxes_2.Position = [313 268 328 200];


% Create UIAxes_3
app.UIAxes_3 = uiaxes(app.UIFigure);
title(app.UIAxes_3, 'Resistance vs Duty cycle')
```

```
xlabel(app.UIAxes_3, 'Duty Cycle %')
ylabel(app.UIAxes_3, 'Resistance ')
zlabel(app.UIAxes_3, 'Z')
app.UIAxes_3.XLim = [0 100];
app.UIAxes_3.XGrid = 'on';
app.UIAxes_3.YGrid = 'on';
app.UIAxes_3.Position = [313 39 328 206];


% Create StartButton
app.StartButton = uibutton(app.UIFigure, 'push');
app.StartButton.ButtonPushedFcn = createCallbackFcn(app, @StartButtonPushed, true);
app.StartButton.Position = [29 70 100 22];
app.StartButton.Text = 'Start';


% Create DutyCycleLabel
app.DutyCycleLabel = uilabel(app.UIFigure);
app.DutyCycleLabel.HorizontalAlignment = 'right';
app.DutyCycleLabel.Position = [7 29 86 22];
app.DutyCycleLabel.Text = 'Duty Cycle (%)';


% Create DutyCycleEditField
app.DutyCycleEditField = uieditfield(app.UIFigure, 'numeric');
app.DutyCycleEditField.ValueChangedFcn = createCallbackFcn(app,
@DutyCycleEditFieldValueChanged, true);
app.DutyCycleEditField.Position = [108 29 100 22];


% Create ResumeButton
app.ResumeButton = uibutton(app.UIFigure, 'push');
app.ResumeButton.ButtonPushedFcn = createCallbackFcn(app, @ResumeButtonPushed, true);
app.ResumeButton.Position = [167 70 100 22];
app.ResumeButton.Text = 'Resume';


% 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 = app1
```

```
% Create UIFigure and components
createComponents(app)


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


if nargout == 0
clear app
end
end


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


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