Lab Sheet

IA 3018 – Data Acquisition Systems

Department of Instrumentation and Automation Technology
University of Colombo

Practical 03

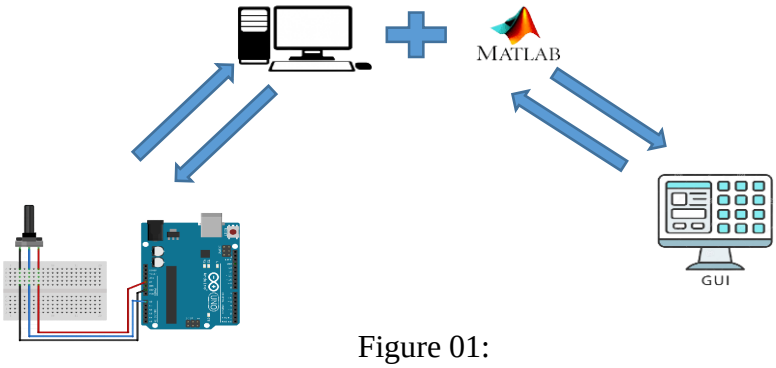

Figure 01:

<u>Part 01</u> – Read an analog data (Real time data) using Potentiometer with Arduino and MATLAB

1. Add the Arduino support package to MATLAB.
2. Connect the Arduino board to computer and record the COM port and board name.(Nano, UNO, Mega, Due)
3. Check the Arduino support package are working in MATLAB using above recorded data in part 2.
4. Create the circuit as figure 02.

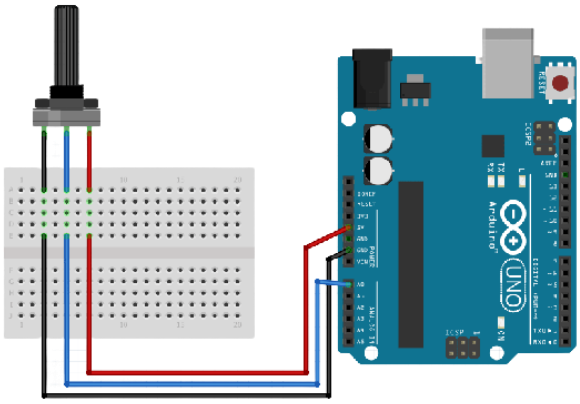

Figure 02: Circuit diagram

5. Write a MATLAB code for display the voltage value.

```
>> readVoltage(a,'A1')

ans =

    3.2307

>> readVoltage(a,'A1')

ans =

    1.9892

>>
```

Figure 03: Example code

6. Writ a MATLAB code using loop (For, While) for plot the graph which is Voltage vs Time as bellow figure 05.

```
>> for i = 1:inf
v(i) = readVoltage(a,'A1');
plot (v);
pause(0.2);
end
Warning: Too many FOR loop iterat
```

Figure 04: Example code

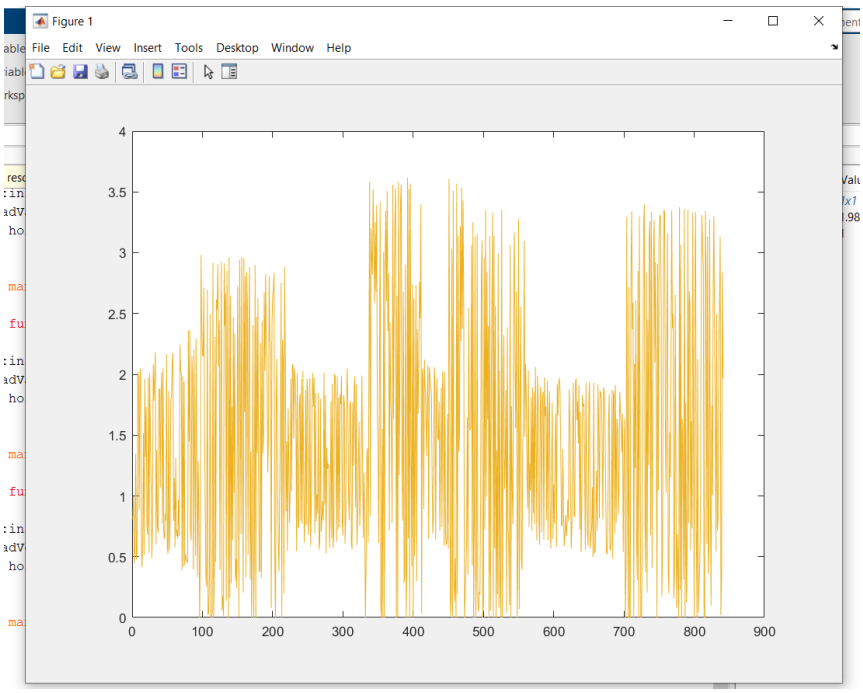

Figure 05: Graph, voltage vs time

7. Adjust the potentiometer and check the graph variations.

<u>Part 02</u> – Read an analog data using potentiometer and Arduino IDE

1. Write an Arduino code for read the analog data and monitor the voltage value.

```
int readValue;
float voltage;

void setup() {
  // put your setup code here, to run once:
Serial.begin(9600);
pinMode (A1,INPUT);
}

void loop() {
  // put your main code here, to run repeatedly:

readValue = analogRead(A1);
voltage = (5./1023.)*readValue ;
Serial.println(voltage);
delay(100);
}
```

Figure 06: Arduino code

2. Upload the Arduino code to Arduino UNO board and check the code.
3. Open the Serial Monitor and check the output voltage values as figure 07.

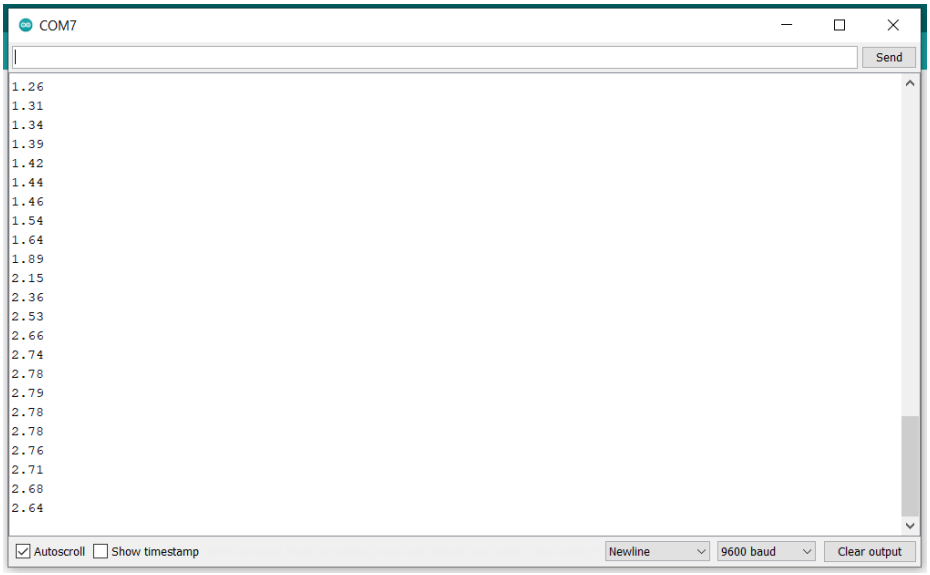

Figure 07: Arduino serial monitor

<u>Part 03</u> - Create a GUI for displaying Voltage value values on Matlab

1. Open GUI window in Matlab.
2. Select blank GUI and save as "Potentiometer".
3. Add topic for the GUI using "Text" function and change the appearance.
4. Add another two "Text" function as figure 08.

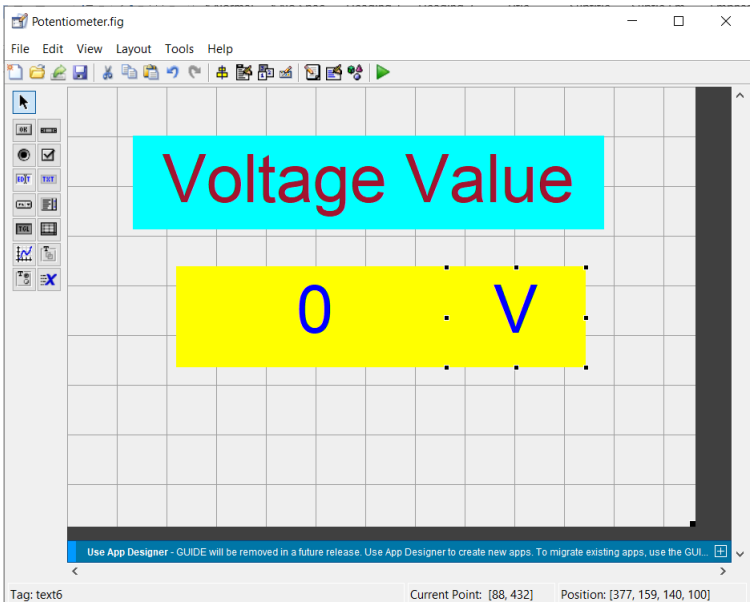

Figure 08: GUI editor

5. Record the Baud rate of Arduino COM port using Device Manger setting in computer.
6. Edit the GUI script using bellow (Figure 09 and Figure 10) commands.

Editor - E:\Matlab\Example\Practical 03\Potentiometer.m

Potentiometer.m

```
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to Potentiometer (see VARARGIN)

% Choose default command line output for Potentiometer
handles.output = hObject;

% Delete any opened ports in MATLAB
delete(instrfind)
% Create a Serial Object
handles.ser = serial('COM7', 'BaudRate',9600,'Terminator','LF',...
    'Timeout',10);
% Associate Serial Event, whenever Terminal Character is recived
handles.ser.BytesAvailableFcn = {@SerialEvent, hObject};
% Open Serial Port
fopen(handles.ser);

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes Potentiometer wait for user response (see UIRESUME)
```

Figure 09: Script part 01

```
% Get default command line output from handles structure
varargout{1} = handles.output;

function SerialEvent(sObject, eventdata, hGui)
% get the updated handle
handles = guidata(hGui);
% get data from serial port
tmp_c = fscanf(sObject);
set(handles.Voltage, 'String', tmp_c)
% Updates handle structure
guidata(hGui, handles)


% --- Executes when user attempts to close figure1.
function figure1_CloseRequestFcn(hObject, eventdata, handles)
% hObject    handle to figure1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
fclose(handles.ser);
delete(handles.ser);
% Hint: delete(hObject) closes the figure
delete(hObject);
```

Figure 10: Script part 02

7. Run the scrip and check the output.

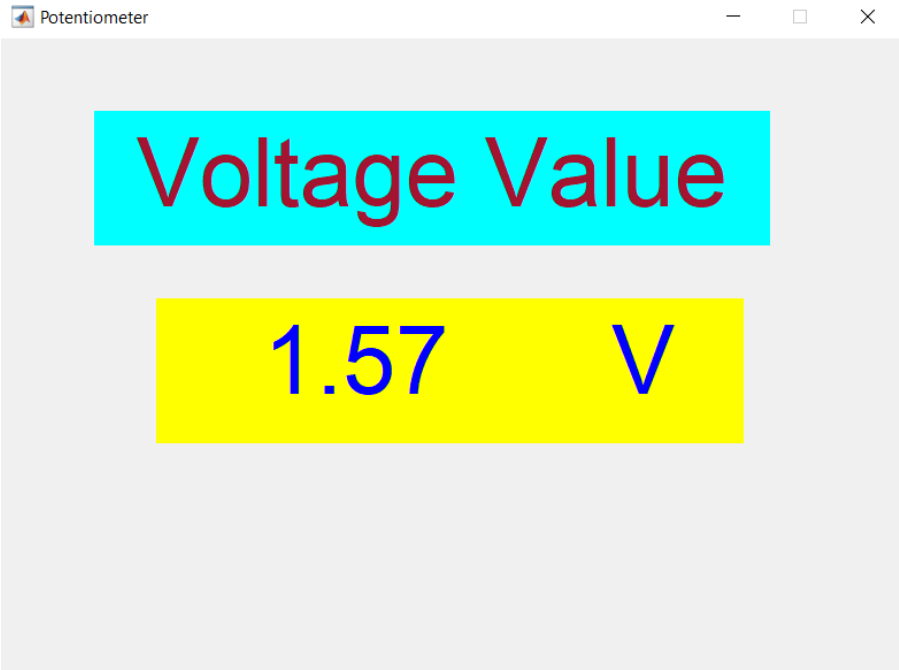

Figure 11: GUI output window