Mini Projects

EEE3096S

Embedded Systems II

University of Cape Town

1 Overview

The mini-project is a chance for you to use what you have learnt in this course and test your problem-solving skills. The projects use everything learnt in the previous practicals - so here's hoping you paid attention and remembered those! All students need to do Mini Project A. Only CS students need to do Mini Project B.

1.1 Scenario

A biology student has approached you with the task of monitoring their private greenhouse. They request that you monitor time of day, time since the system has been running, light levels, temperature, and humidity. They have a device that monitors humidity. It outputs an analog voltage between 0 and 3.3V. For testing and development purposes, you decide to use a potentiometer to mimic the humidity sensor. With the experience you've gained working with the Raspberry Pi over the past few months, you decide that this is a good chance to test your newly acquired knowledge and establish an effective development environment for further projects.

The biology student has a computer in their greenhouse that they use to play music to their

plants as a part of their research, so they are able to access the Raspberry Pi on location through a terminal or VNC. However, they wish to be able to also monitor the data remotely. They say their younger brother, who plays with Arduinos, has spoken a lot about Blynk for IoT devices. You decide to look into these options for remote monitoring.

The also request that your device output a voltage to their own custom recording equipment. The voltage should be calculated as follows:

$$V_{out} = \frac{LightReading}{1023} \times HumidityVoltageReading$$

If the output voltage goes under 0.65V or over 2.65V, an alarm should sound. A button should be added to dismiss the alarm, and the alarm should only sound if it the previous alarm dismissal was more than 3 minutes ago (i.e. the alarm can only sound every 3 minutes).**The three minute timer should start when the alarm is sounded, not when the alarm is dismissed.**

1.2 Basic Details

Your objective for Mini Project A and B is to design an environment logger. An environment logger interacts with the world around it by measuring any number of factors from GPS location to air pollution. While a single sensor is useful to monitor a single location, many can be scattered around a broader area to monitor that area as well as any patterns emerging in that broader area.

Project A and B differ in how that data is presented to end users. Project A will require use of a single Raspberry Pi, which relays data to and is controlled via an app on your phone created through Blynk.

Project B requires two Raspberry Pis. Instead of using Blynk, one Pi works as the environment sensor. It will send the gathered data using a protocol called MQTT to a node red server, which will be hosted on the second Raspberry Pi. Users should be able to log in to a hot spot hosted on the second Raspberry Pi and access the Node Red server to see the data being captured.

The objective of these mini-projects is to get some experience on what it might be like to create and deploy a real IoT device - what the pracs in the course have been leading up to!

2 Mini-Project A

This project examines ELO5.2, which requires the student to develop a representative embedded system. It requires hardware/software interfacing, so an ADC is sampled by software, and processing operations are applied to read the data.

2.1 Pre-Project Requirements

- You will need to have completed all the previous practicals in order to complete this project.
- Using the design flows presented in class, we strongly recommend beginning by drawing up an initial design and mapping the required tasks to (at the least) a simple timeline or Gantt chart.

2.2 Outcomes

In addition to creating a working system, you will learn about the following in this project:

- ADC - MCP3008
- Temperature sensor - MCP9700A
- IOT - Blynk

2.3 Deliverables

At the end of this practical, you must:

- Demonstrate your working implementation to a tutor through a formal presentation. This will take the form of a formal presentation in a 10 minute slot you will book in advance. See Table II for details on the demonstration.
- Submit your report to Vula, in the format shown in Table I
- For information on the marks and what sections to cover, refer to the marking guide in Section 2.9

2.4 Hardware Required

- All hardware from previous pracs
- Potentiometer to mimic VPD sensor
- LDR
- ADC
- Temperature Sensor

2.5 System Overview

Figure 1 shows the system overview for the mini project.

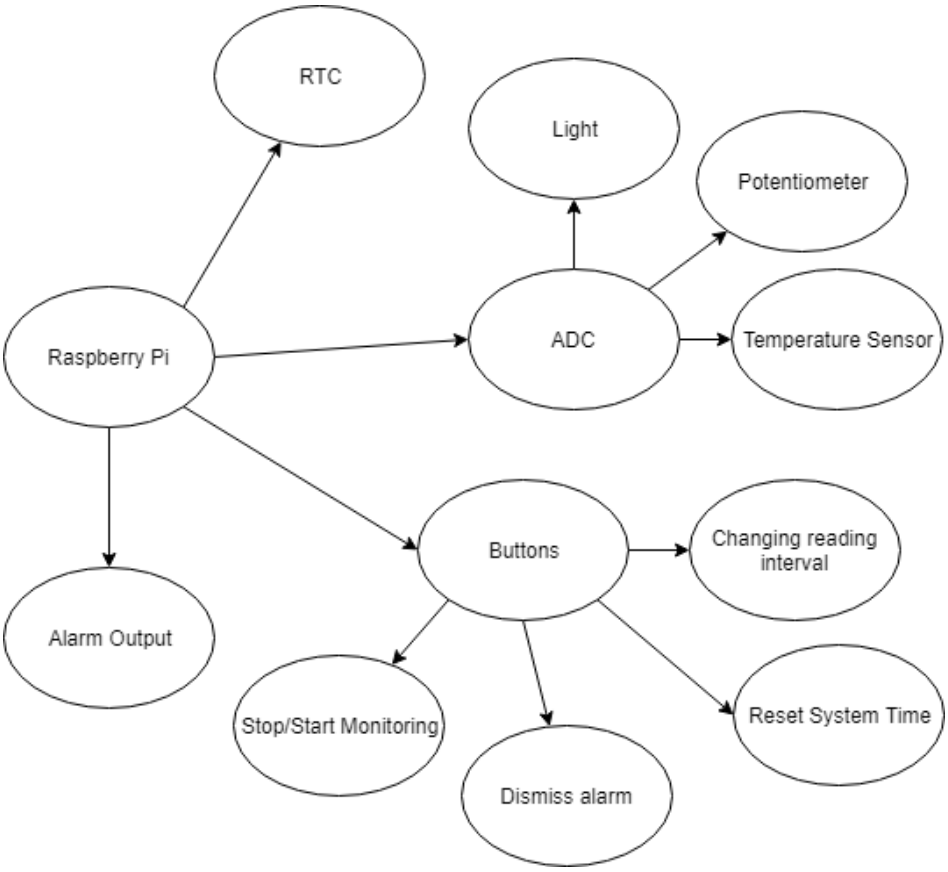

Figure 1: Components of the Mini-Project

Figure 2 shows an example of what the app you create in Blynk might look like. As you can see there's a terminal for accessing the output of the print statements, three values as read from the ADC, and an indicator to indicate if the alarm has gone off. There are various widgets in Blynk. It's suggested you play with your energy budget to develop the most intuitive design and aesthetically pleasing design possible.

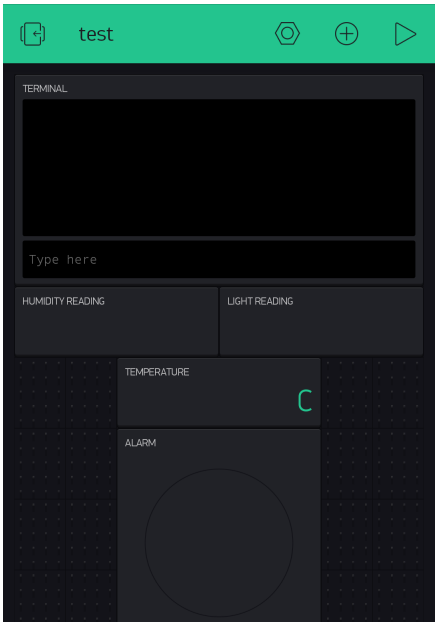

Figure 2: Example Blynk Project

2.6 Hardware requirements

You will need to build your design and demonstrate it to the marker.

2.7 Software requirements

You need to do the following:

- Set up the RTC. You can use either the Raspberry Pi's supplied kernel driver for an RTC, or you can interface with it as you did in Practical 3.
- Create a thread for reading from the ADC. The value read from the temperature sensor needs to be converted to degrees Celsius. See the datasheet (linked above) for the formula.
- Create interrupts for all the button functionality (don't forget to debounce your inputs)
- Output the correctly calculated voltage over the DAC.
- Create an output signal to notify the biology student that there is an issue. You can flash an LED using a PWM signal, you can play audio through the audio jack, or you can buy a buzzer from WhiteLab.
- In the main() loop, print values to the screen as described in Section 2.8

2.8 Description

- The system should have hard-coded thresholds. When a value read by the device goes over a threshold, an alarm should sound. A button must be pressed to dismiss the alarm.
- By default, the system continuously monitors the sensors every 1s using this format:

RTC Time	Sys Timer	Humidity	Temp	Light	DAC out	Alarm
10:17:15	00:00:00	0.5 V	25 C	595	0.29V	*
10:17:20	00:00:05	1.5 V	25 C	595	0.87V	*
10:17:25	00:00:10	1.7 V	25 C	595	0.98V	
10:17:30	00:00:15	2.2 V	25 C	782	1.68V	
10:17:35	00:00:20	3.3 V	25 C	998	3.22V	

  Notice how the alarm flag is still active at 10:17:20, as it is only dismissed before the reading at 10:17:25. Also note that even though the DAC out value exceeds the alarm threshold at time 10:17:35, no alarm is sounded as the last alarm was sounded less than three minutes prior.

- The reset switch resets the system timer and cleans the console

- The frequency switch changes the frequency of the monitoring. The possible frequencies are 1s, 2s and 5s. The frequency must loop between those values per event occurrence.
- The stop switch stops or starts the monitoring of the sensors. The system timer is not affected by this functionality.
- Blynk must be used to:
  - View live logging information (System time and ADC values)
  - Be notified of alarms

2.9 Marking Guide

Table I: Project A marking Guide

Heading	**Report**	**Demo**
Introduction	Provide a short introduction ($\sim \frac{1}{2}$ page) to your project explaining your main design choices and how the report has been structured (15 marks)	Introduce yourselves and your project. [8 marks]
Requirements	The requirements section should provide a refined UML Use Case diagram of the system, according to your implementation, and any accompanying text that is needed to clarify the requirements. Highlight any departures or additions that you may have made compared to the original project description given in this document. [15 marks]	Have an (at least draft) UML Use Case to show the tutor. Show this briefly, indicating any departures/additions. [6 marks]
Specification and Design	This section should provide a UML State Chart describing the main operation. Add a UML class or deployment diagram (or other suitable diagram) to indicate the structuring of your implementation (e.g. code modules/classes you may be using). You dont need to provide fine detail of the system, the diagram(s) can be e.g. at the level of functions. You should also include a circuit diagram [20 marks]	Briefly show your design, you need not show more than one rough diagram (e.g. draft state chart) to the tutor. [10 marks]

Implementation	This section should give some snippets of important code and explanations for this (or referring to particular functions in code files). The point here is elaborating any parts of the State Chart that are not so straightforward to turn into code. [20 marks]	You should have a code file open already (e.g. where the main function is) before you start the demo. Briefly confirm to the tutor that this is part of the program that will be demoed. [6 marks]
Validation and Performance	Provide at least a paragraph or two explaining the performance of the system. A snapshot could be included and you could show test cases where you have tested that the system works reliably (e.g. using a powersupply to set the value given to the ADC). [20 marks]	This is a main aspect of the demonstration. See Validation Check Sheet in Section 2.10. [40 marks]
Conclusion	Give a summary of the extent that the system was found to be successful. Discuss if you think that a system working in this way might be considered a potentially useful product. [10 marks	End you demo with a short conclusion, reflect on the activity. [10 marks]
References	Provide a few references if relevant.	Questions [20 marks]
Total	100	

2.10 Project A Validation Sheet

Table II: The Mark sheet used in Demos

Demo Mark Validation Sheet		Marked by:		
Student Numbers:				
Component	**Category**	**Description**	**Max Marks**	**Mark**
Blynk	Connected and responds	Blynk should connect to the Pi (2) and reponds to changes on the Pi (5 - ldr, pot, temp, alarm, start/stop log)	7	
	Design	All features of the EnviroLogger should be reflected in the widgets used in Blynk	5	
		The use of widgets should be aesthetically pleasing as well as intuitive (use of labels, placement of widgets)	2	
ADC	Temperature	Temperature reported on is accurate and responds to input (warming up when sensor is held, etc)	2	
	Potentiometer	Responds to changes input, reading displayed is between 0 and 3.3V	2	
	Light sensor	Voltage divider	2	
		Reports a value between 0 and 1023	2	
Buttons/ Logging	Implementation	Uses interrupts (1 ea) and debouncing (1 ea)	8	
	Interval Reading	Changes between 1, 2 and 5s	3	
	Reset SysTime	Resets system time to 0	1	
	Dismiss Alarm	Clears alarm and outputs message saying so.	2	
	Stop/Start logging	Pauses the print out to system console, but timer continues increasing	1	
Alarm	Implementation	Calculation implemented	1	
	DAC	Outputs correct voltage	1	
	Alarm	Can't sound more often than 3 min intervals	1	
Mark			**40**	

3 Mini-Project B

This project is for CS students only. Of course, if you'd like to attempt, you can.

3.1 Scenario

The biology student appreciates the work you have done developing a rough prototype. Their research has started gaining ground, and more people are becoming interested in their research. Because you did such a good job of developing the base system, the biology student has asked you to develop a means of a system where anyone can access their data. Being

competent developers, well aware of web development, as well as being aware of the world of embedded systems you decide to use MQTT to transfer data. You want to make the system as accessible as possible, so you decide to use a low-cost Raspberry Pi to host a Node-Red server. For the sake of the prototype, you create a hotspot on the Raspberry Pi that people can connect to in order to access the server. You decide on the following design, as shown in Figure 3:

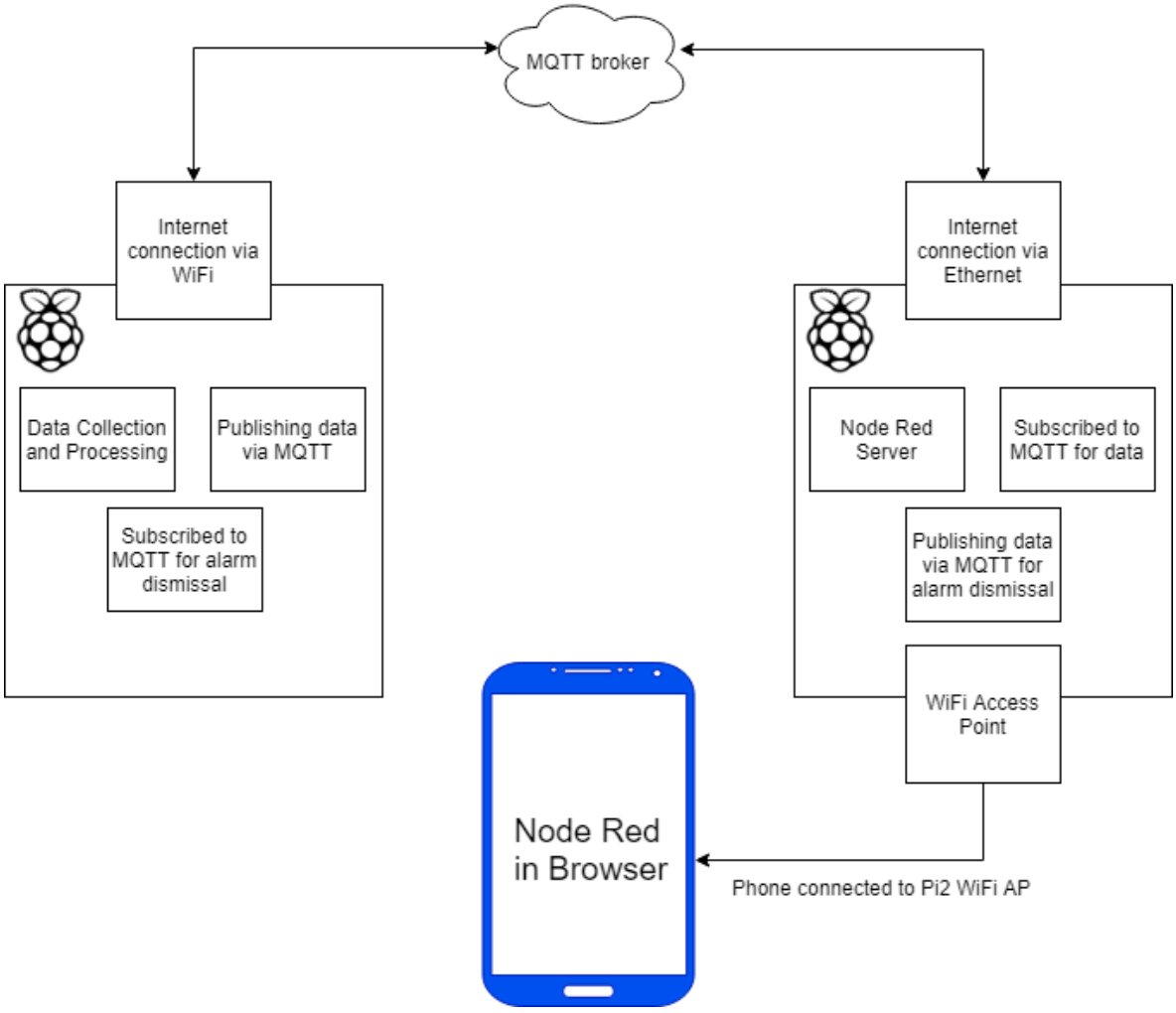

Figure 3: Project B System Overview

3.1.1 Overview

For this task, two Raspberry Pis are required. Both are required to publish and subscribe to data. One Pi must publish the recorded data, and subscribe to an alarm dismissal signal. A second Pi should publish an alarm dismissal message and subscribe to the data collected by the first Pi. A web page must be hosted on the second Pi, and it's WiFi interface should be turned into an access point. Devices should be able to connect to this access point, and

view a web page which can mimic the Blynk App designed in Mini-Project A. For the MQTT broker, you can use a publicly available one, or host one on one of the Pis.

Essentially, this is a repeat of the first project, but using a second Raspberry Pi with MQTT and Node Red as opposed to Blynk.

3.1.2 Outcomes

You will learn about:

- MQTT (It's suggested you read this introduction to MQTT and MQTT on NodeRed
- NodeRed (It's suggested you read Getting Started with NodeRed on the RaspberryPi to get started. There's also a nice plugin for good looking dashboards here. You can also read the Adafruit Guide.)
- Hosting an access point on the Pi. Notes are available in the lab handbook.

3.1.3 Deliverables

At the end of this practical, you must

- Demonstrate your implementation to a tutor. They will log in to your hotspot through their phone, view values, and adjust the alarm threshold.
- Submit a short write up. See the marking guide in section 3.1.6

3.1.4 Hardware Required

You require the hardware from Project A, as well as a secondary Pi.

3.1.5 Software Requirements

- You need to use an MQTT broker to publish and subscribe to messages relating to your practical.
- You need to create a Node Red server that displays your data gathered from the first Raspberry Pi.
- You will also need to have an option to adjust the alarm threshold from the NodeRed server.

3.1.6 Marking Guide

Table III: The Write Up Format For mini-project B

Section of Report	Description	Marks
Introduction	An introduction to what's new in this project	10
Design	Design of the system. Design of the server. Talk about the stack used for development. Include UML and block diagrams, and mention any hardware/software interfacing issues.	30
Implementation/ Build Proces	Some code snippets and steps in building the device and server.Can think of this as a simple methodology.	15
Instructions for use	How to operate the system. You should include some screenshots and photos.	15
Testing/Results	How did you ensure your system works? What did you do to test the functionality, and what were the results? You can include screenshots or photos, but you need to talk about them (it's not enough to just have a photo).	20
Conclusions	How well did your system work? Did you achieve the objectives? What could you do to improve the system?	10
TOTAL		**100**

3.1.7 Project B Validation Guide

Table IV: Project B Demo Marks

ProjB Validation Sheet		Marked By:		
Student Numbers:				
Component	**Category**	**Description**	**Max marks**	**Mark**
Hotspot	Connectivity	Viewable in WiFi Menu from phone	1	
		Tutor can connect to it	1	
Node Red	Implementation	WebPage is accessible	1	
		Starts on Boot	1	
	Design	All features of the EnviroLogger should be reflected in the webpage (5 - LDR, pot, temp, alarm, start/stop log)	5	
		The use of widgets should be aesthetically pleasing as well as intuitive (use of labels, placement of components)	3	
		Can adjust the alarm threshold (1) and is tested (2)	3	
Questions			5	
Marks Obtained			**20**	