

Stellenbosch University

Electrical & Electronical Engineering Department

Remote monitoring of a coffee percolator with a web-based platform using sensors and an actuator

Student
Izak Jean Britz

Study Leader
Prof. G-J van Rooyen

17 June 2014

Report submitted in partial fulfilment of the requirements of the module Project (E) 448 for the degree Baccallaureus in Engineering in the Department of Electrical and Electronic Engineering at the University of Stellenbosch

Declaration

I, the undersigned, hereby declare that the work contained in this report is my own original work unless indicated otherwise.

Signature *Date*

..../...../.....

Acknowledgements

I would like to thank Prof. G-J van Rooyen for guiding me during the project and providing a comfortable workspace in the MIH Media Lab.

I also want to thank MTN and Trintel Telecommunications who supported this project with their financial assistance.

Lastly I want to thank my parents for their support throughout the semester when I needed it most.

Summary

The aim of this project was to investigate which factors influence the operation of a percolator. Cost-effective components such as sensors and transducers were used to measure temperature, water levels and power consumption. An TRIAC circuit was used to control the power delivered to the percolator by sending a command from a web-based platform. A monitoring system was developed where sensor data was collected from analog-to-digital converters and then transmitted to the Trinity SMART Sight platform using a GSM modem provided by Trintel. RFID tags were used to track the total amount of coffee being consumed. A graphical web-based user interface was developed for the end-user to monitor the current state of the coffee percolator.

Opsomming

Die doel van hierdie projek was om faktore te identifiseer wat die werking van 'n koffiemasjien beïnvloed. Koste-besparende komponente soos sensors en "transducers" was gebruik om temperature, watervlak en kragverbruik te meet. 'n TRIAC–stroombaan was gebruik om die kragtoevoer na die koffiemasjien te beheer. Dit is uitgevoer deur 'n bevel te stuur van die Trintel platform. 'n Moniterings sisteem was ontwikkel om sensor data te vergader van analoog-na-digitale omsetters en is dan na die Trinity SMART Sight platform gestuur via 'n GSM-modem verskaf deur Trintel. RFID plakkers is gebruik om kontrole te hou oor die mate van koffieverbruik. 'n Grafiese web-gebaseerde paneel is ontwikkel vir die verbruiker om die koffiemasjien mee te moniteer.

Contents

1 Introduction **1**
- 1.1 Problem statement . . . 1
- 1.2 Project description . . . 1
- 1.3 Objectives and limitations . . . 2
  - 1.3.1 Objectives . . . 2
  - 1.3.2 Limitations for the scope of the project . . . 2

2 Background information **4**
- 2.1 Coffee percolator . . . 4
  - 2.1.1 Coffee making cycle . . . 5
  - 2.1.2 General coffee making rules . . . 6
- 2.2 Trinity Telecommunications . . . 6
  - 2.2.1 Sierra Wireless AirVantage asset model . . . 7
  - 2.2.2 Trinity SMART Sight . . . 7
  - 2.2.3 Sierra Wireless Airlink modem . . . 8
  - 2.2.4 AT commands . . . 8
  - 2.2.5 AWTDA extended commands . . . 9
- 2.3 Temperature sensors . . . 9
  - 2.3.1 Thermistor . . . 9
  - 2.3.2 Microchip MCP9700 active thermistor . . . 10
  - 2.3.3 Melexis MLX90614 infrared thermometer . . . 10
- 2.4 Coffee level measurement . . . 11
  - 2.4.1 eTape liquid level sensor . . . 11
  - 2.4.2 Ultrasonic transducers . . . 12
  - 2.4.3 Webcam with image processing . . . 13
  - 2.4.4 Force-sensing resistor . . . 13
- 2.5 Current sensors . . . 13
  - 2.5.1 Current transformer . . . 13
  - 2.5.2 Hall effect sensor . . . 14
- 2.6 Relays . . . 14
  - 2.6.1 Electromechanical relay . . . 15
  - 2.6.2 Triode for alternating current . . . 15
- 2.7 AC to DC rectifier circuit . . . 17
- 2.8 Radio-frequency identification . . . 17
- 2.9 Arduino Mega 2560 . . . 18

2.9.1 Arduino processing 18

3 System Design **19**
3.1 Overall system design 19
3.1.1 Sensor placement on percolator 20
3.1.2 Asset model 22
3.1.3 Flow of data 22

4 Detail design **23**
4.1 Hardware 23
4.1.1 Arduino 23
4.1.2 Temperature sensors 24
4.1.3 Water level measurement 25
4.1.4 Coffee percolator 27
4.1.5 TRIAC driver circuit 27
4.1.6 DC power supply 29
4.1.7 Current sensor 30
4.1.8 Voltage sensor 30
4.1.9 RFID payment system 31
4.2 Details of the software 31
4.2.1 Dashboard overview 33

5 Measurements and Tests **34**
5.1 Testing of GSM modem communication 34
5.1.1 Hypothesis 34
5.1.2 Test design 34
5.1.3 Results 35
5.2 Testing of the temperature sensors 36
5.2.1 Initial conditions 36
5.2.2 Hypothesis 36
5.2.3 Test design 36
5.2.4 Results 36
5.3 Test of the ultrasonic sensor 37
5.3.1 Hypothesis 37
5.3.2 Test design 37
5.3.3 Results 37
5.4 Test of voltage sensor to measure the mains voltage 38
5.4.1 Hypothesis 38
5.4.2 Test design 38
5.4.3 Results 38
5.5 Test of the RFID payment system 39
5.5.1 Hypothesis 39
5.5.2 Test design 39
5.5.3 Initial conditions 39
5.5.4 Test design 39

5.5.5 Results . . . 39
5.6 Test of the TRIAC control circuit connected to the percolator . . . 41
5.6.1 Hypothesis . . . 41
5.6.2 Test design . . . 41
5.6.3 Results . . . 41

6 Conclusions **42**
6.1 Objectives achieved . . . 42
6.2 Future work . . . 43

Appendices **47**

A Project planning schedule **48**

B Project specification **50**

C Outcomes compliance **52**

D Additional tables **54**

E Additional program flow diagrams **56**

F Bill of materials **63**

G Source code **65**
G.1 Git repositories . . . 66
G.2 External Arduino source code . . . 66

H Additional diagrams **67**

I Additional circuit diagrams **69**

J Interconnections between components **72**

K Test instructions **75**
K.1 Test instructions for the temperature sensors test . . . 76
K.2 Test instructions for RFID payment system . . . 77
K.3 Test instructions for testing the TRIAC control circuit . . . 77
K.4 Test instructions for the voltage sensor . . . 78
K.5 Test instructions for testing the ultrasonic sensor . . . 79

L Additional test data **80**
L.1 Ultrasonic sensor's test data . . . 81
L.2 Current sensor's test data . . . 81

M Test screenshots **82**
M.1 Test results with testing the TRIAC switching circuit 82
M.2 Test results of the voltage sensor 82
M.3 Test results of the RFID payment system 84
M.4 Test results of ultrasonic sensor for measuring the reservoir's water level . . . 85

N Test logs **86**
N.1 Terminal output when executing the RFID payment system test 87

List of Figures

2.1 Different components found in a coffee percolator [1]. 5
2.2 Screenshot of SMART platform's gadget maker[5]. 7
2.3 Serial link implementation for a 2-wire connection[7]. 8
2.4 (a) Melexis MLX90614 TO-39 package. (b) Suggested 2-wire serial interfacing schematic [12]. 10
2.5 eTape liquid level sensor [14]. 12
2.6 Principle of an active sonar [15]. 12
2.7 Measuring the liquid level within a coffee pot using a webcam. 13
2.8 The configuration of a toroidal current transformer to measure current [19]. . 14
2.9 Basic configuration of an electromechanical relay [20]. 15
2.10 (a) Symbol for an TRIAC. (b) The physical structure of an TRIAC. (c) The TRIAC's four quadrants of operation [21]. 16
2.11 (a) Modified Miller Encoding. (b) Manchester Encoding [23]. 18

3.1 Overall system diagram. 20
3.2 Placement of sensors on percolator with the RFID payment system feature. . 21

4.1 The interface circuits of the temperature sensors with the Arduino. 24
4.2 The placement of the ultrasonic transducers on the coffee percolator. 26
4.3 Configuration for using the ultrasonic transducers to measure the water level [26]. 26
4.4 The Safeway coffee percolator. 27
4.5 Interfacing circuit where the opto-coupled TRIAC is used to drive the main TRIAC. 28
4.6 Interfacing circuit for the Arduino where a voltage transformer is used to measure the mains voltage connected to the percolator. 31
4.7 Output sinusoidal waveform of the voltage transformer with mains voltage as input. 31
4.8 Screenshot of dashboard. 33

5.1 Screenshot of dashboard after the test has finished. 37

A.1 Project planning schedule using a Gantt chart. 49

E.1 Flow control diagram of main program. 58
E.2 Flow control diagram for the "serial1_event"-method. 59
E.3 Flow control diagram of the "sendATCommand"-function. 60

E.4 Flow control diagram of how unsolicited responses from the SMART platform are handled. 61
E.5 Flow control diagram for processing the TriggerPulse command. 62

H.1 The flow of data within the system 68

I.1 A full-wave rectifier circuit. 70
I.2 Circuit designed to be used to build a DC power supply. 70
I.3 The circuit for converting the reflected signal to be compatible with the ADC port on the Arduino. 70
I.4 The circuitry within the percolator. 71

L.1 Measurement of reflected signal's peak voltage when a flat surface is place within 3 centimeters of the ultrasonic transducers. 81
L.2 Measurement of reflected signal's peak voltage when the water resevoir is empty. 81
L.3 50 Hz signal measured by the current sensor. 81

M.1 Screenshot of the oscilloscope with the TRIAC switching with 10% duty cycle. 83
M.2 Screenshot of the oscilloscope with the TRIAC switching with 30% duty cycle. 83
M.3 Screenshot of the oscilloscope with the TRIAC switching with 50% duty cycle. 83
M.4 Screenshot of the oscilloscope with the TRIAC switching with 70% duty cycle. 83
M.5 50 Hz signal sampled by the Arduino. 84
M.6 Screenshot of the coffee usage gadget after the first test for the RFID reader is executed. 84
M.7 Screenshot of the coffee usage gadget after the second test for the RFID reader is executed. 84
M.8 Plot of ADC values when the water reservoir is empty. 85
M.9 Plot of ADC value when the water reservoir has 465 ml of water. 85
M.10 Plot of ADC value when the water reservoir has 920 ml of water. 85
M.11 Plot of ADC value when the water reservoir is full of water. 85

List of Tables

3.1 Data variables defined in asset model. 21
3.2 A Command defined in asset model. 22
3.3 Operating states defined in asset model. 22

4.1 Specifications taken into consideration for interfacing the Arduino and the Melexis MLX90614 infrared thermometer. 25
4.2 Summary of component's voltage and current requirements to be used for designing the DC power supply with. 29

5.1 Test AT commands sent to the GSM modem via the Arduino. 34
5.2 Output of ultrasonic sensor versus amount of water in reservoir 38
5.3 Voltage sensor results 38
5.4 List of RFID-cards' UID used in this test 39
5.5 Initial conditions for RFID test 39

C.1 ECSA outcomes criteria. 53

D.1 Features of the Arduino Mega 2560. 55

F.1 Bill of materials for this project. 64

J.1 Analogue pin connections between various sensors and the Arduino Mega 2560. 73
J.2 General purpose I/O port connections with various components in the system. 74

K.1 The first sequence in which the cards are swiped pass the RFID reader. . . . 76
K.2 The second sequence in which the cards are swiped pass the RFID reader. . 77

Nomenclature

AC	Alternating Current
ADC	Analog–to–Digital Converter
ASCII	American Standard Code for Information Interchange
AVR	A modified HarVard architecture 8–bit RISC single chip micro–controller
BJT	Bipolar Junction Transistor
CRC–8	8–bit Cyclic Redundancy Check
CT	Current Transformer
DC	Direct Current
DCE	Data Communication Equipment
DSP	Digital Signal Processor
DTE	Data Terminal Equipment
EEPROM	Electrically Erasable Programmable Read–Only Memory
EMI	Electro–Magnetic Interference
FSR	Force–Sensing Resistor
GSM	Global System for Mobile communications
GUI	Graphical User Interface
I^2C	Inter-Integrated Circuit
IC	Integrated Circuit
IDE	Integrated Development Environment
IR	InfraRed
LED	Light–Emitting Diode

M2M	Machine–To–Machine
PCB	Printed Circuit Board
PEC	Packet Error Code
PWM	Pulse–Width Modulation
RAM	Random-Access Memory
RF	Radio–Frequency
RFID	Radio–Frequency IDentification
RMS	Root Mean Squared
SCL	Serial Clock Line
SCR	Silicon–Controlled Rectifier
SDA	Serial Data Line
SMBus	System Management Bus
SMS	Short Message Service
SPI	Serial Peripheral Interface bus
TRIAC	TRIode for Alternating Current
UART	Universal Asynchronous Receiver and Transmitter
UID	Unique IDenitification number

Thyristor terminology

Symbol	Terminology	Description
I_{GT}	Gate trigger current	The maximum value of gate current required to switch the device from the off-state to the on-state under specified conditions.
I_H	Holding current	The minimum current that must be flowing (MT1 & MT2; cathode and anode) in a thyristor to keep the device in a regenerative on-state condition. Below this holding current value the device will return to a blocking state, off condition.
I_L	Latching current	The minimum current that must be applied through the main terminals of a TRIAC (or cathode and anode of a thyristor) in order to turn from the off-state to the on-state while its I_{GT} is being correctly applied.
dv/dt	Critical rate of rise of off-state voltage	Also, commonly called static dv/dt. It is the minimum value of the rate of rise of forward voltage which will cause switching from the off-state to the on-state with gate open.

Chapter 1

Introduction

In the fast-paced working environment most employees are accustomed to a cup of coffee to boost their energy levels when getting tired. However this habit causes large quantities of coffee being wasted and unnecessary appliances left switched on. Therefore this project is aimed to address the issue by developing a monitoring system to regulate the coffee percolator.

1.1 Problem statement

The aim of this project is to investigate which factors influence the operation of a percolator. Inexpensive components such as sensors, transducers and actuators could be introduced to monitor the percolator from a web-based platform. Data derived from these factors will be integrated into a software program.

1.2 Project description

A percolator must be monitored by using inexpensive sensors to indicate its current operating status. Sensors should measure the heating element's temperature; the steam temperature; the coffee pot's temperature and coffee level; the level of the water reservoir and power consumption of the percolator. These sensors must be connected to a microprocessor in such a way that the microprocessor can periodically update and store their outputs. After updating the sensor's data, it must then be processed and sent to the Trinity SMART Sight platform using an GSM modem. The Trinity SMART Sight platform is also referred to as the SMART platform. The percolator must be controlled remotely using an actuator that has parameters to turn the percolator on or off when the command is sent from the SMART platform. End users should be able to make payments, in the form of tokens, with an Radio–Frequency Identification-tag, better known as RFID-tag, when they pour coffee. The entire system must be compatible with the household's 240V mains power outlet.

The project is provided with an GSM modem from Trinity Telecommunications, also known as Trintel, which they have set up over the MTN cellular network. All telemetry data will be processed by the SMART platform and displayed on the platform's web-based user interface, which can be accessed by anyone that has login credentials to the website.

1.3 Objectives and limitations

1.3.1 Objectives

The objectives set for this project are:

1. Obtain and integrate a set of cost-effective sensors that measures the following:
    - Base temperature - the temperature that the heating element emits.
    - Steam temperature - the temperature of the steam passing through the tube to the shower head.
    - Coffeepot temperature - the temperature of the coffeepot, when it is placed on the base plate.
    - Coffeepot water level - the amount of coffee in the coffeepot.
    - Water reservoir level - the level of water in the reservoir.
    - Power consumption - the amount of power being consumed by the system over a specified time frame.
2. The measurements of the sensors listed above must be converted into a data format, which can be tracked and presented on the SMART platform.
3. The percolator's power must be controlled using an actuator. A command, sent from the SMART platform, should be assigned for the actuator that instructs it to switch the power, connected to the percolator, on or off.
4. There must be a simple payment system in place, which accepts RFID cards, to keep track of the amount of cups of coffee being poured.
5. The components' cost used for this project should be kept at a minimum and must not alter the aesthetic value of the percolator when it is being integrated.
6. The system is only allowed to use the mains power supply as a source. Therefore a direct current (D)C power supply is required for supplying the microprocessor and the GSM modem with power.

1.3.2 Limitations for the scope of the project

The following limitations has been put in place to narrow the focus and aim of the project:

- The system is unable to replace the used coffee filter.
- The system is also unable to clean itself after each usage.
- The water reservoir must be filled up manually, when empty. (The system may alert the consumer with an short message service (SMS) or email message that it is empty).
- The system is not supplied with a backup power supply in case a power failure occurs.

In the following chapters different approaches are examined for obtaining these objectives. Then an overall system design is done to incorporated each objective and finally the objectives are implemented using a combination of hardware and software designing. Tests are then performed to see if these implementations done beforehand are capable of achieving its corresponding objective.

Chapter 2

Background information

For developing a system which monitors a percolator, it is necessary to know the basic functionality of a percolator. When brewing the coffee there are some general recommendations to take note of in order to improve the usage of coffee being brewed.

This project is required to use the services of Trinity's SMART Sight. This service is powered by AirVantage M2M cloud, which allows developers to create asset models. These models are integrated with the SMART platform, which has a web-based developing environment. The SMART to configure their asset models.

Temperature is the measurement of heat. Heat is either transferred by convection, radiation or conduction. As convection is irrelevant when dealing with a percolator, measurement of temperature would only be possible due to radiation or conduction. Ways of measuring water level must be investigated. Possible options is by either physically or indirectly using the laws of physics. Methods of measuring the alternating current (AC) voltage and current is also investigated to calculate power consumption. Means of controlling the percolator by switching it on or off is possible using a relay. Compare different types of relays by considering the pro's and con's against cost-effectiveness. Building a direct current(DC) power supply one needs to apply the neccessary theoretical knowledge of rectifying an AC signal to DC. As specified in the project description a RFID reader must be used to implement a simple payment system. All of these various components needs a central micro-processor in order to function properly, therefore it is required to select this component with utmost certainty.

2.1 Coffee percolator

[1] gives a more detailed explanation of how a coffee percolator works, but a more condensed version is given here.

The percolator consists of seven components:

1. The glass coffee pot (see Figure 2.1(a)) where the coffee accumulates when the water has exited the filter.

2. The reservoir (labelled as 1 on Figure 2.1(b)) where water is poured in at the start of the coffee-making cycle.

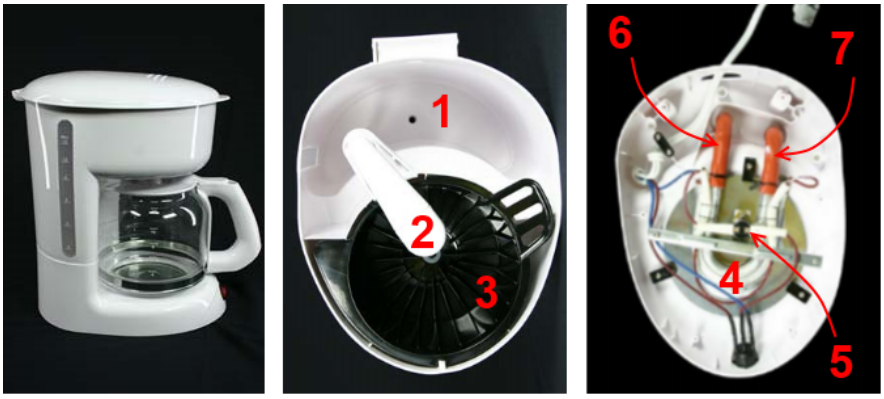


Figure 2.1: Different components found in a coffee percolator [1].

3. The shower head (labelled as 2 on Figure 2.1(b)) where the water arrives from the hot-water tube and is sprayed onto the coffee grounds.

4. The funnel (labelled as 3 on Figure 2.1(b)) where the coffee grounds are placed within a paper filter.

5. The heating element (labelled as 4 on Figure 2.1(c)) which provides heat when water arrives from the reservoir.

6. The heat switch (labelled as 5 on Figure 2.1(c)) which regulates the temperature of the heating element. It will terminate the power supply when the heating element has reached a specific maximum temperature.

7. The one-way valve. This component is not visible on these figures, but it will become clear later where it is situated on the percolator.

The above-mentioned resistive heating element is lodged between the aluminium tube and the base plate with heat-conductive paste. This paste ensures that heat is efficiently transferred to the aluminium tube and that the base plate keeps the coffeepot warm. The same paste attaches the heat switch to the heating element ensuring good contact to regulate the temperature of the element.

2.1.1 Coffee making cycle

To start the coffee making cycle one must fill the reservoir with adequate water and place fresh coffee grounds in the paper filter. The paper filter should be placed in the funnel. As the heating element warms up, it causes the water to flow from the reservoir through the hole (next to label 1 in Figure 2.1(c)) and into the orange tube.

Between the reservoir and the orange tube there is an one-way valve connected, which allows water to flow (due to gravity) into the aluminium tube. When the heating element has accumulated enough heat, the boiling water in the aluminium tube is forced out of the reservoir and into the shower head connected via the white tube.

The one-way valve prevents boiling water flowing back into the reservoir. Therefore, the boiling water can only flow upwards through the white tube into the shower head. For this to happen, a narrow tube has to be used, which causes large bubbles to carry a column of water upwards gurgling into the shower head.

Water flows through the shower head and sprays onto the coffee grounds. While this sprayed water oozes through the paper filter, it picks up the oil essence from the coffee grounds and carry it into the coffee pot.

2.1.2 General coffee making rules

Some general recommendations listed in [2] and [3] apply to each coffee making cycle to ensure the best taste:

- Coffee should be brewed for 4.5 to 5 minutes using a ratio of 55 grams of ground coffee per liter of filtered water.
- Filtered water or spring water is recommended, because tap water imparts off flavours to the coffee while some minerals are essential to the coffee flavour.
- Distilled water is not recommended for brewing coffee as it lacks the minerals to bring out the natural flavours of the coffee.
- Coffee grounds should be removed directly after coffee is made. Seepage from coffee grounds imparts the good flavour and aroma.
- Keep prepared coffee at a temperature of 80°C to 88°C until served.
- Never allow more than an hour to elapse between making and serving the coffee.

2.2 Trinity Telecommunications

This project is provided with the services of Trinity Telecommunications, specifically the Trinity SMART Sight service. This is a suitable solution for this project, because it allows data to be transmitted onto their cloud platform. Using Sierra Wireless GSM modem one can transmit and receive telemetric data remotely. The SMART Sight service uses the AirVantage M2M cloud asset models to define the a telemetric data structure to represent the variables that your application wants to monitor. Other metrics include commands, events, alarms and operating states which can also be defined in your asset model. Trinity also provides the user with a web-based graphical user interface (GUI), where data variables and commands can be interface with preloaded gadgets. These gadgets is then placed onto a dashboard, which is effectively the end user's GUI. For this project the dashboard helps the end user to monitor an asset more effectively by quantifying the data. For instance the temperature variable values can be collected over a certain time period and can then be plotted as a line graph. This dashboard, located at `smart.trintel.co.za`, is available via any device with a web browser given that the user is provided with login credentials.

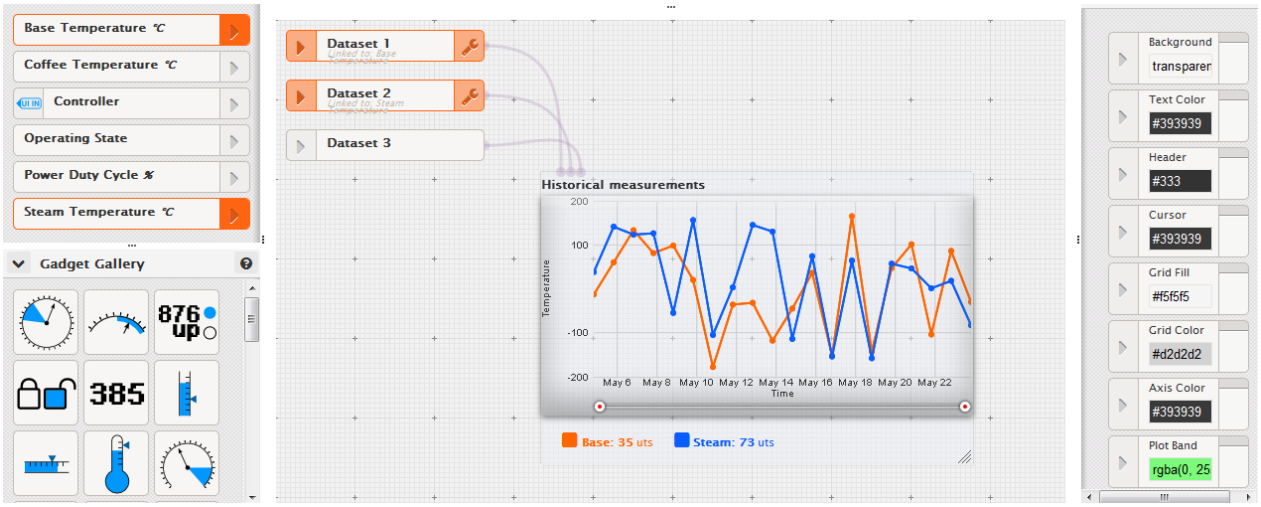


Figure 2.2: Screenshot of SMART platform's gadget maker[5].

2.2.1 Sierra Wireless AirVantage asset model

The AirVantage M2M Cloud is a service that allows developers to create an asset model using the Eclipse-based AirVantage Configuration Tool. In the context of this project, the asset would be the percolator. The asset model contains the following metrics:

- Data — Variables which are subject to change when given the appropriate command
- Commands — Messages being sent from the platform to the asset, which will trigger the pre-configured action on the asset.
- Events — Pre-defined message sent from the asset to the platform to trigger an alert at the end user's side.
- Alarms — They are the same as events, except they send the alert message via email or SMS.
- Operating states — Pre-defined states, which are dependent of the state of the physical device being monitored. For example if the percolator is switched off, its operating state would be updated by transmitting the message that the current state is 'switched off' to the GSM modem.

Sierra Wireless programmable modems works out of the box with this service [4]. In this project we can directly deploy our models on the AirVantage operating portal hosted at `platform.trintel.co.za` by Trintel.

2.2.2 Trinity SMART Sight

Trinity's SMART Sight platform is a web-based tool which can be configured to retrieve and display specified data [5].

It allows the developer to specify, consolidate and present data on a GUI. The collection of data is structured according to a AirVantage asset model. These metrics set within the

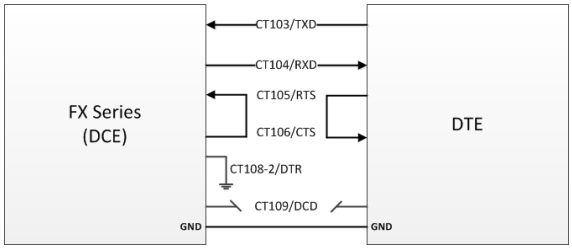


Figure 2.3: Serial link implementation for a 2-wire connection[7].

asset model can then be used to monitor an application via a dashboard, through the use of web gadgets as shown in Figure 2.2. These gadgets can take the form of gauges, graphs and dials, to provide a graphical "snapshot" of relevant information [5].

2.2.3 Sierra Wireless Airlink modem

Trinity Telecommunications provided this project with an Sierra Wireless Airlink FXT009 GSM modem. The FXT009 provides a highly extendible platform for the creation of M2M solutions[6].

This programmable modem has an RS-232 serial port, which can interface with other devices as shown in Figure 2.3. A set of AT commands is provided on the modem's firmware. For these two devices to communicate with each other one device should be the Data Communication Equipment (DCE), while the other one is the Data Terminal Equipment (DTE). Usually the DTE device is the computer and the DCE is a modem or another carrier-owned service. In our case the FXT009 would be the DEC and the Arduino would be the DTE.

2.2.4 AT commands

AT commands allows developers to extract information (e.g. signal strength or network reception) or to configure settings situated on the modem's firmware.

AT commands use the American Standard Code for Information Interchange (ASCII) to represent their commands. AT commands are typically used in telecommunication to transfer data and to execute predefined operations on a modem device. When commands are executed from a DTE the response is also in ASCII format, e.g. "OK". AT commands listed in [8] can be transmitted via the Arduino to the GSM modem acting as the DCE. The GSM modem also supports the Open AT Application Framework, a software package that allows the user to extend the modem's capability by programming a set of custom AT commands for specific applications. Most of the commands are to change internal registers on the modem, but there are commands that execute external operations such as making a phone call or sending an SMS. Commands that change internal register values usually have four types of commands, which are distinguishable by a specific syntax.

2.2.5 AWTDA extended commands

Trintel developed their own extended AT command set using the Open AT Application Framework that enables data communication between the FXT009 modem and their M2M platform. Data can be sent and received by the modem via the online platform. Commands with parameters can be sent from the platform to the modem with the modem having to send an acknowledgement back to the platform to indicate that it is completed. An event sent from the modem to the platform can trigger an SMS or email notification. These events are connected to each asset model that contains a valid email address or phone number of the owner of the respective asset.

2.3 Temperature sensors

There is a wide range of temperature sensors available on the market. For this project two types of temperature sensors were needed: one that measures the relative temperature of the medium or object closest to it and one that measures temperature by measuring the blackbody radiation being emitted by the heat of the object.

2.3.1 Thermistor

Thermistors "is a type of resistor whose resistance varies significantly with temperature, more than in standard resistors [9]." They are classified as follows:

- positive temperature coefficient (PTC): the resistance increases with increasing temperature
- negative temperature coefficient (NTC): the resistance decreases with increasing temperature

In [10] the relationship is discussed between the resistance and the temperature of thermistors. The thermistor's temperature can be derive from Eq.(2.1), which requires the resistance of itself. Thermistor can be calibrated at three known temperature points to determine the coefficients A, B and C through solving three equations, with the form of Eq. (2.1), simultaneously. This equation is better known as the Steinhart–Hart equation [10]:

$$\frac{1}{T} = A + B\ln(R) + C(\ln(R))^3 \tag{2.1}$$

T is the temperature variable, while the R represents the resistance of the thermistor at temperature T. A minimum of three resistance values where the temperature is known are required for solving the three coefficients. Sometimes the manufacturer supplies a lookup table with known temperature values mapped to a single resistance value. Using this table one can obtain the temperature using a thermistor without using Eq.(2.1).

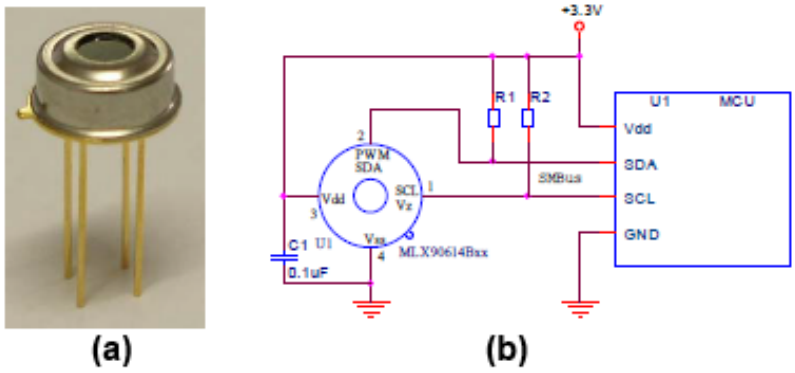


Figure 2.4: (a) Melexis MLX90614 TO-39 package. (b) Suggested 2-wire serial interfacing schematic [12].

2.3.2 Microchip MCP9700 active thermistor

The Microchip MCP9700 active thermistor can measure temperatures within the range of -40°C to 125°C, however it has an accuracy of 2°C within the range of 0 to 70°C. This active thermistor does not require any additional signal-conditioning circuit. This eliminates the demand of a biasing circuit or extra software overhead, which a passive thermistor does need to operate properly. The voltage output pin (V_{OUT}) can be directly connected to any microprocessor's analog-to-digital converter (ADC) input with an 8-bit resolution or more [11]. For example, if an 8-bit ADC port is connected to the active thermistor, then the resolution would be 1°C per bit.

2.3.3 Melexis MLX90614 infrared thermometer

The MLX90614 is an infrared thermometer for non-contact temperature measurements. Both the IR sensitive thermopile detector chip and the signal conditioning application specific standard product (ASSP) are integrated in the same TO-39 package. Due to its low noise amplifier, 17-bit ADC and its digital signal processor (DSP) unit, a resolution of 0.02°C is achieved. The calculated temperature of the objects with its field-of-view and ambient temperatures are available on the embedded random-access memory (RAM). They are accessible via a 2-wire serial SMBus compatible protocol or via the 10-bit pulse-width modulated (PWM) output of the device.

The MLX90614 is factory calibrated for ambient temperatures within the range of -40°C and 125°C. It can also detect temperatures from objects within the range of -70°C and 382°C. The average temperature is calculated within the field-of-view of the sensor with a typical accuracy of ±0.1°C.

An optical filter (high-pass filter) that cuts off the visible and near infrared radiant flux is integrated in the package to provide ambient and sunlight immunity. The wavelengths that are allowed to pass through are optical filter ranges from 5.5 to 14 micrometers [12].

System management bus

The System Management Bus (SMBus) is a two-wire interface through which simple system and power management related chips can communicate with the microprocessor using Inter-Integrated Circuit (I^2C), because SMBus is based on the principals of I^2C [13]. SMBus also use two lines for communication; a serial data line (SDA) and a serial clock line (SCL) that have an open drain or open collector in order to perform the wired-AND function.

Version 1.1 of the SMBus specification introduced an optional packet error checking mechanism to improve reliability and communication robustness. It can easily be implemented by appending an Packet Error Code (PEC) at the end of each message transfer. The PEC uses an 8-bit cyclic redundancy check (CRC-8) with each read or write bus transaction to calculate the PEC. The PEC may be calculated using the polynomial in Eq. (2.2) This PEC must be calculated in the order of the bits received, excluding the ACK-, NACK-, START- and STOP-bit. This means that the PEC is computed over the entire message from the first START condition.

$$C(x) = x^8 + x^2 + x + 1 \tag{2.2}$$

One advantage of the SMBus over I^2C is that it allows slave receivers to acknowledge their own address always as a mechanism to detect a removable device's presence on the bus. Unlike I^2C, the SMBus has a built-in timeout which is triggered when SCL is low for longer than 35 milliseconds. One trade-off due to this timeout is that it cannot guarantee complete compatibility with the I^2C bus with a clock frequency greater than 100kHz [13].

2.4 Coffee level measurement

Three methods of measuring the liquid levels are considered to achieve the objectives stated in section 1.3. Two different methods are considered to measure the level of water in the reservoir, while the third and only method considered for measuring the coffee pot's level is by using a force-sensing resistor (FSR). By measuring the weight of the coffee first, the volume of the coffee can be derived with basic geometry given the dimensions of the coffee pot.

2.4.1 eTape liquid level sensor

The eTape Liquid Level Sensor from Milone Technologies is a variable resistor that varies its resistance linearly inversely proportional to the level of the water it is immersed in.

It is a solid-state, continuous fluid level sensor for measuring levels on water, non-corrosive water-based liquids and dry fluids (powder).

The eTape sensor is manufactured using printed electronic technologies which employ additive direct printing process to produce functional circuits [14]. When the sensor is immersed into a liquid the sensor's envelope is compressed by the hydrostatic pressure of the fluid in which it is immersed. This results in a change in resistance (typically 60 Ω per centimeter), which corresponds to the distance from the top of the sensor. The sensor comes prefabricated with four pre-soldered wires, which can be used to interface the sensor with a microprocessor using a simple voltage dividing scheme.

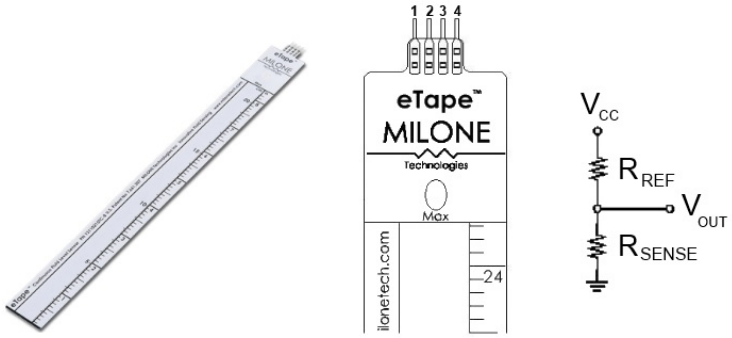


Figure 2.5: eTape liquid level sensor [14].

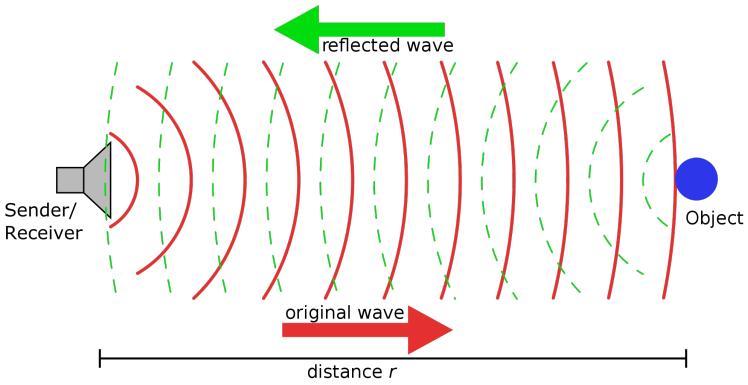


Figure 2.6: Principle of an active sonar [15].

2.4.2 Ultrasonic transducers

Ultrasound is an oscillating sound pressure wave with a frequency greater than the upper limit of the average human hearing range [15]. On the frequency spectrum ultrasound is situated approximately at 20 kHz and upwards. Ultrasound may be generated by an ultrasonic transducer, and can be used to detect objects and measure distances.

The ultrasonic transducers investigated here contain a small piezoelectric component, which resonates and emits a sound signal according to the frequency of the input signal. An ultrasonic transducer can be either a transmitter or a receiver. These transducers are popular for measuring distance from a fixed position, because it can implement the principle of an active sonar.

This principle is implemented as follows: Typically a short pulse of ultrasound is emitted from the transmitter. When there is an obstruction in the path of the ultrasound pulse, some portion of the transmitted ultrasound pulse is reflected and the receiver can then interpret this echoed sound wave. By measuring the elapsed time between the sending and receiving of the signal along with the knowledge of the speed of sound in the medium, the distance between the receiver and the object can be calculated. Figure 2.6 illustrates how the sonar principle is used with the ultrasonic transducers to measure distance.

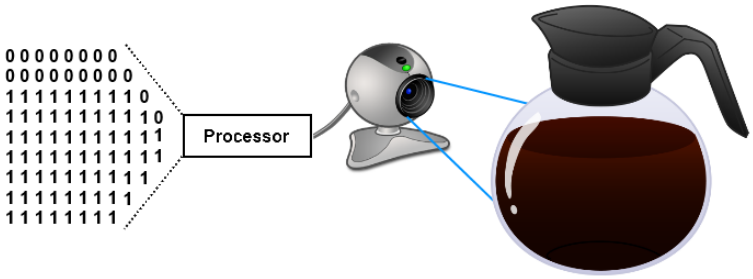


Figure 2.7: Measuring the liquid level within a coffee pot using a webcam.

2.4.3 Webcam with image processing

The coffee pot's coffee level can be measured with a webcam by detecting the dark liquid against a lighter background. Figure 2.7 shows how a stationary webcam at a fixed distance from the coffee pot can measure the liquid level. One can implement a simple algorithm which classifies each pixel, based on their RGB value, as one ('1') if it is close to being black otherwise it is zero ('0'). The microprocessor can typically store the classified pixel values within an array. This array should be at least a column vector with a total length equal to the coffee pot in centimeters. If the microprocessor has enough memory available then the array can be expanded to more than one column with rows equal to the length of the coffee pot in millimeters, for instance, if more accuracy is required. Taking the mean value between the columns for each row will eliminate sporadic errors that may be caused by small movements captured on the webcam. The level of the coffee pot is then calculated by summing the mean value over each row.

2.4.4 Force-sensing resistor

A force-sensing resistor (FSR), consists of a conductive polymer, which changes resistance in a predictable manner following application of force to its surface. As with all resistive based sensors, force-sensing resistors require a relatively simple interface and can operate satisfactorily in hot environments. Compared to other force sensors, the advantages of FSR's are their size (thickness typically less than 0.5 mm), low cost and good shock resistance [16].

2.5 Current sensors

"A current sensor is a device that detects electrical current (AC or DC) in a wire, and generates a signal proportional to it "[17]. The output signal would typically be an analogue voltage that varies in the range of -5 volts to 5 volts. We will only cover the current transformer sensor and the Hall effect sensor.

2.5.1 Current transformer

The current transformer (CT) method of operation is based on Faraday's law of induction that states "the induced electromotive force (emf) around a closed path is equal to the negative rate of change of the magnetic flux with respect to time passing through the area

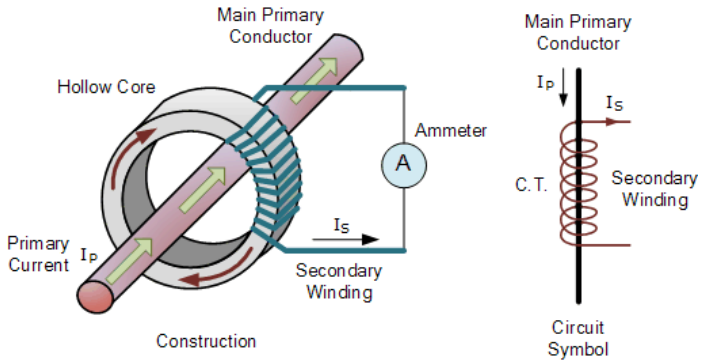


Figure 2.8: The configuration of a toroidal current transformer to measure current [19].

enclosed by the path [18]". Current and voltage can be measured by placing the current-carrying conductor through the hollow core of a toroidal inductor. The configuration shown in Figure 2.8 makes it possible to measure current by using passive, inexpensive components (e.g. a 100 Ω resistor) connected in parallel to the secondary windings.

2.5.2 Hall effect sensor

The basic principle of the Hall effect is that, when a perpendicular magnetic field is present, an Lorentz force is exerted on the current. This force disturbs the current distribution, resulting in a potential difference (voltage) across the output. This is known as the Hall voltage. The interaction of the magnetic field and the current is shown in Eq. (2.3).

$$V_H = I \times B \tag{2.3}$$

Hall effect sensors are generally expensive, but have very good accuracy and minimal software and hardware overhead to measure the current passing through the sensor.

2.6 Relays

The relay is "a device that provides an electrical connection between two or more points in response to the application of a control signal [20]". In this section, the electromechanical relay and the family of thyristors, particularly the triode for alternating current (TRIAC), are discussed as possible solutions to control the percolator. Different aspects of the opto-coupled TRIAC are addressed, which must be taken into account when designing the driver circuit for controlling the main TRIAC. The opto-coupled TRIAC's purpose (also called the driver TRIAC) is to be used in conjunction with the main TRIAC, which is connected in parallel with the AC power supply and the load. This opto-coupled TRIAC enforces a physical isolation between the microprocessor and the AC power. Now the AC load can be driven with a programmable timer signal generated by the microprocessor, which is required for this project.

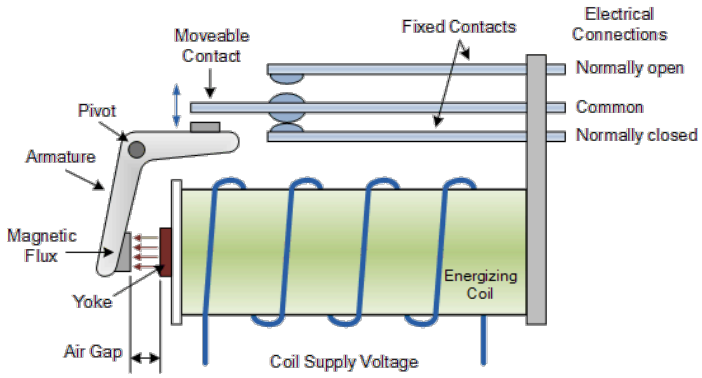


Figure 2.9: Basic configuration of an electromechanical relay [20].

2.6.1 Electromechanical relay

Electromechanical relays is a basic way to use magnetic force to control a mechanical switch. These relays are usually enclosed within a container to isolate them from any electrical interference. The 'load currents' which are typically switched ranges from a few amperes up to 20 amperes.

Figure 2.9 illustrates the switching mechanism of this relay. This mechanism is described by [20] as follows:

> *...electromechanical relay consists of an energising coil called the "primary current" wounded around an iron core. This iron core has both a fixed portion called the 'yoke', and a moveable spring-loaded part called the 'armature'. rThe 'armature' completes the magnetic field circuit by closing the air-gap between the fixed electrical coil and the moveable 'armature'. The 'armature' is hinged or pivoted allowing it to freely move within the generated magnetic field closing the electrical contacts that are attached to it [20].*

The disadvantage of these relays are when the switch is closed and the circuit is conduction there is always a certain amount of "contact resistance", which leads to unnecessary power loss. As the contact strips starts to wear off they will start to show signs of arcing damage if they are not properly protected form high inductive or capacitive loads [20].

2.6.2 Triode for alternating current

For a better understanding of how an triode for alternating current or an TRIAC operates, it is useful to know the basic behaviour of an silicon controlled rectifier (SCR), because the analysis of an TRIAC is essentially two parallel SCR's orientated in opposite directions [21].

The TRIAC is essentially a semi-conductor switch mainly used for switching AC power sources or for controlling AC motors. Just like a SCR, the TRIAC is triggered into conduction by applying a voltage signal to its gate and MT1 as labelled in Figure 2.10. The difference with the TRIAC is that it conducts current in either directions, while the SCR only conducts current in one direction like a diode. The TRIAC also differs from the SCR in that either a positive or negative gate signal will trigger the TRIAC into conduction [21].

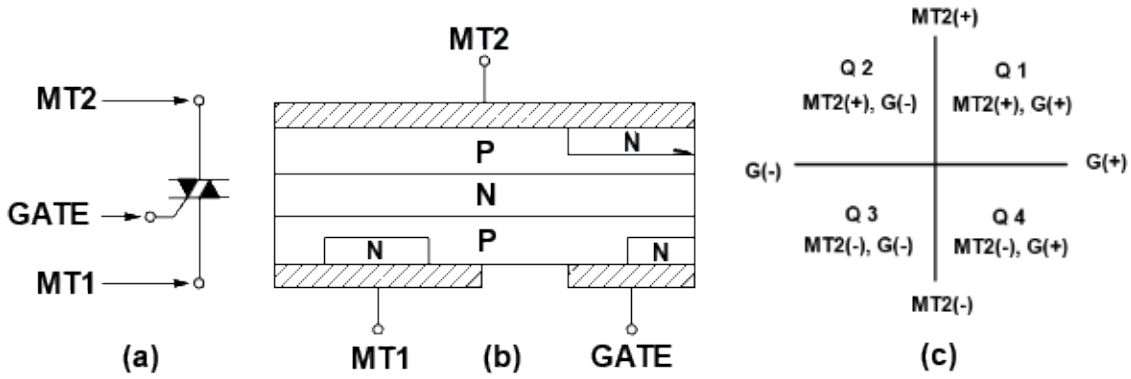


Figure 2.10: (a) Symbol for an TRIAC. (b) The physical structure of an TRIAC. (c) The TRIAC's four quadrants of operation [21].

Since the TRIAC is conducting current in both ways, the terms "anode" and "cathode" used for the SCR has no meaning. Therefore the current-carrying terminals of the TRIAC are labelled as MT1 and MT2. The "GATE" in Figure 2.10(b) remains the gate terminal. To avoid confusion, it has become standard practice to specify all currents and voltages using MT1 as the "neutral" point [21]. Figure 2.10(b) shows the basic structure of a TRIAC. It is a five-layer device with the region between MT1 and MT2 an P-N-P-N switch (SCR) in parallel with an N-P-N-P switch (complementary SCR). Figure 2.10(b) above gives some insight into the TRIAC's ability to be triggered with either a positive or negative gate signal.

> *The region between MT1 and GATE consists of two complementary diodes. A positive or negative gate signal will forward-bias one of these diodes causing the same behaviour as a SCR. This action breaks down the blocking junction regardless of the polarity of MT1. Current flow between MT2 and MT1 then causes the device to provide gate current internally. The device will remain on until this current flow is interrupted [21].*

Ideally the TRIAC will conduct only when it operates in the first or third quadrant.

Snubber circuit

Snubber circuits consists out of a capacitor and a resistor which are usually connected in parallel with the TRIAC's MT1 and MT2 terminals. They are compulsory for inductive loads, but in this project the load is purely resistive. Therefore the snubber circuit is only used for preventing the TRIAC to start conducting due to other loads causing voltage spikes on the AC line. For the TRIAC it is also important to keep the rise of voltage (dv/dt) within the recommend range. This dv/dt is limited by adding an snubber across the TRIAC. The capacitor will then limit the dv/dt across the TRIAC, while the resistor limits the surge of current from the capacitor when the TRIAC conducts [21].

Controlling methods

There are two different methods of control namely, phase control and cycle control.

Using the phase control method, it allows the TRIAC to only conduct a pre-defined portion of the AC signal. Typically it would conduct the specified portion over each half cycle. This has the disadvantage of generating unwanted electro-magnetic interference (EMI) [21]. Also this method does not permits the usage of an opto-coupled TRIAC for interfacing between the main TRIAC, connected to AC power, and a microprocessor.

Cycle control, on the other hand, is a method where an opto-coupled TRIAC is used for controlling AC power delivered to a load. This method allows logic circuits and microprocessors to easily control AC power by giving the TRIAC driver on-off or burst of DC voltage signal, for example a PWM signal. This method can be used for both the zero-crossing and non-crossing TRIAC drivers.[21]

Zero-point switching within opto-coupled TRIAC

Zero-point circuits have the function of toggling their output every time its input waveform crosses zero voltage. These circuit's switching characteristics are highly desirable in many applications, because they do not generate EMI when used with high AC voltages. A zero-point switch controls sine-wave power in such a way that either complete cycles or half cycles of the AC power supply are applied to the load. This type of switching is primarily used to control power delivered to resistive loads such as heaters [21].

2.7 AC to DC rectifier circuit

Direct-current power supply is generally obtained from an AC source. The rectification of the AC signal is done using four diodes connected according to Figure I.1 to form a diode bridge. The signal is then further rectified by using a capacitor that induces a ripple voltage. This ripple voltage starts to "droop" when the load requires more current, therefore a voltage regulator is required to compensate for the "droop".

2.8 Radio-frequency identification

Radio-frequency identification (RFID) is the wireless non-contact use of radio-frequency (RF) electromagnetic fields to transfer data, for the purposes of automatically identifying and tracking tags attached to objects [22].

In this project the ISO/IEC 14443 Type A RFID booktags and cards are used, because it is inexpensive and the booktags can be glued onto coffee mugs. The card's loop antenna typically communicates within 10cm of the reader's antenna at a radio-frequency of 13.56 MHz.

When the card is placed within the reader's RF field the reader device inductively couples with the card's antenna, which then supplies the card with power. The reader then amplitude modulates the RF field to send information to the card. The card contains a demodulator to convert the amplitude modulation to digital signals. It also contains a clock extraction circuit that produces the 13.56 MHz digital clock for use within the card. The data from the reader is clocked in, decoded, and processed by the card's integrated circuit [23].

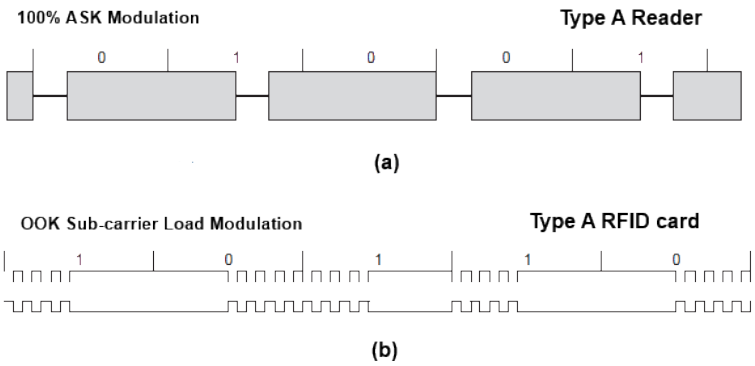


Figure 2.11: (a) Modified Miller Encoding. (b) Manchester Encoding [23].

The Type A card uses 100 percent amplitude modulation of the RF field for communication from the reader to the card with modified Miller encoded data — see Figure 2.11(a). Communications from the card to reader uses on/off keying modulation (OOK) of an 846.5 kHz sub-carrier with Manchester encoded data — see Figure 2.11(b) [23] In this project each RFID-card is identified by reading its unique identification number (UID). The cards used for this project has four unsigned bytes, which can take on $2^{(8\times4)}$ different values as the UID.

2.9 Arduino Mega 2560

Arduino is an open–source development board. The Arduino products are based on modified Harvard architecture 8-bit RISC single chip microprocessors, better known as an AVR.

The Arduino Mega 2560 is am microprocessor development board, which uses Atmel ATmega2560-chip. [24]. This board has stackable headers for interfacing with additional external modules. This enables the developer to extend the capability of this board. The Arduino Mega supports many low-level communication protocols. such as UART, Serial Peripheral Interface (SPI), I^2C and has also a built-in USB interface (using the ATmega16u2 chip[24]) to program or debug the microprocessor using the Arduino Integrated Development Environment (IDE). This board can either be powered via USB or by using the power jack. Table D.1 lists some attractive features of the ATMega2560-chip. Features shown in Table D.1 were considered during the component selection phase.

2.9.1 Arduino processing

Arduino has simplified the software coding of their products by developing libraries based on C/C++ to make it easier for their users to develop projects. For instance, they developed a library (called SoftwareSerial) to implement UART communication via software. It has two main functions: set-up and loop. The set-up function is where all the pin modes and voltage levels are set and where the hardware and software serial ports baud rate are typically set. The loop function can be considered as an infinite while loop, just like a state-machine. The Arduino IDE does not have an extensive debugging facility, but most libraries have incorporated debug traces by printing certain statements to the hardware serial port.

Chapter 3

System Design

The proposed system should be built from inexpensive electronic components where possible. Components such as sensors, actuators, a radio-frequency identification (RFID) reader, an AC-to-DC rectifier circuit and a microprocessor are required for developing the system. The different functions, listed in section 1.3 that the sensors must provide have to be accurate to $\pm 2°C$ and must have minimal software and circuit overhead. Software overhead must always be taken into account when working with microprocessors, such as the Arduino, because it has significantly smaller memory capacity than the average desktop computer. The current consumed by sensors must also be kept to a minimum to preserve it for the microprocessor usage, which normally has a higher demand for it.

The only actuator used in this project is a relay circuit for controlling the percolator. Since the system uses AC power, and is controlled by a microprocessor using DC power, one should ensure that these two different power sources are electrically isolated from each other.

The choice of which microprocessor to use is important for this project, because it will determine whether every component has a port to integrate with it. Aspects such as the number of ADC ports and timer circuits were taken into account during the component selection of the microprocessor. An Arduino Mega 2560 microprocessor board was decided on as it meets the system requirements of this project and also provides support for the commercially-off-the-shelf RFID reader shield being used.

There are no restrictions on what sensors, transducers or actuators may be used, except for the GSM modem. A preconfigured GSM modem was provided by Trintel including a registered MTN SIM card. The modem uses two AT command sets. One set is provided with the modem's firmware and the other extended set was provided by Trintel and specifically developed to communicate with their SMART platform. This chapter provides information on the different possible components and implementations that were considered.

3.1 Overall system design

The overall system consists of various components as shown in Figure 3.1. The modem and the Arduino communicate through an UART serial connection that sends metric data and receives commands to and from the SMART platform respectively. The RFID reader communicates with the Arduino through its SPI-interface, which triggers an interrupt on

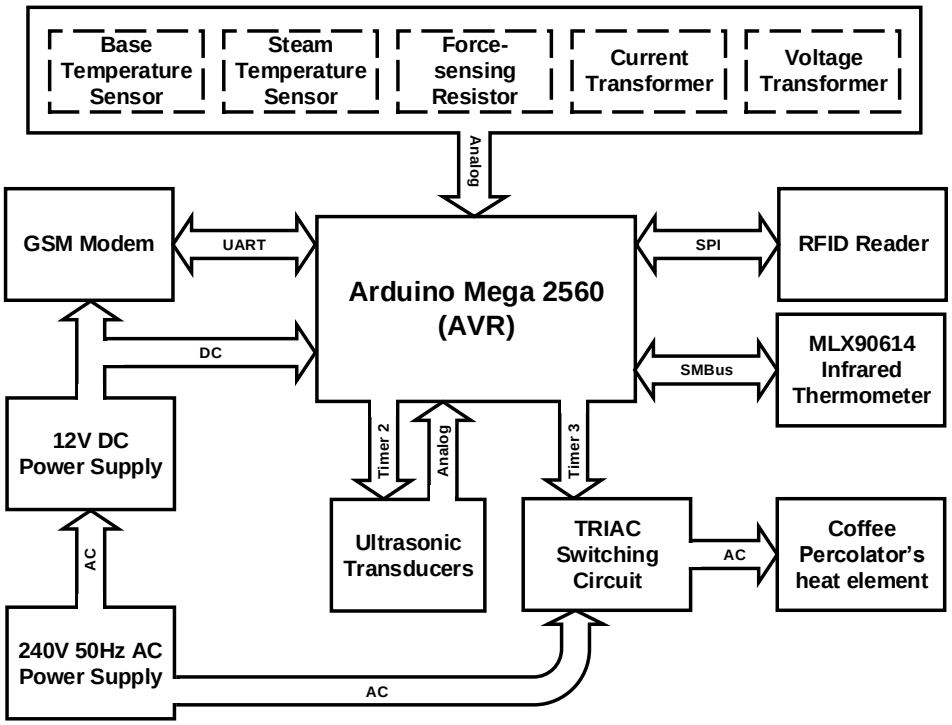


Figure 3.1: Overall system diagram.

the Arduino whenever an ISO14443-compliant card is present within the RFID reader's RF field. Most of the sensors are connected to the ADC ports on the Arduino, except for the MLX90614 infrared thermometer which communicates with the Arduino via its I^2C-interface. The ultrasonic transducers uses Timer 2 in fast PWM mode to emit short burst of square waves. The only component which will receive commands from the SMART platform is the TRIAC switching circuit that controls the AC power given to the percolator's heating element. The 12 DC power supply does not communicate with the Arduino or the GSM modem. It only supplies them with power.

3.1.1 Sensor placement on percolator

In section 2.1 is the basic functionality of the percolator discussed. Based on this and the objectives specified in 1.3 sensors were placed on the percolator as shown in Figure 3.2. The RFID payment system is not fool prove, it is only a feature that has been put in place for the user to log its coffee usage on the system. The responsibility of paying of the coffee lays between all the end users that uses this system.

Figure 3.2 also indicates the flow of water through the system using bold arrows. The blue squares indicates where temperature sensors are placed, while the green and red rectangular shapes indicates the placement of the sensor used for measuring the water reservoir level and the sensor for measuring the coffee pot level respectively. The microprocessor is placed further from the percolator to prevent coffee being spilled onto the reader.

Table 3.1: Data variables defined in asset model.

Variable name		Type	Unit		Variable name		Type	Units
temp	base	integer °C		rfid	uid1	string		
	steam	integer	°C			uid2	string	
	coffee	integer	°C			uid3	string	
level	resevoir	double	mm			uid1_cup_usage	int	cups
	coffee	double	mm			uid2_cup_usage	int	cups
power	voltage	double	V			uid3_cup_usage	int	cups
	current	double	A		misc	currentState	int	
	duty_cycle	integer	%					

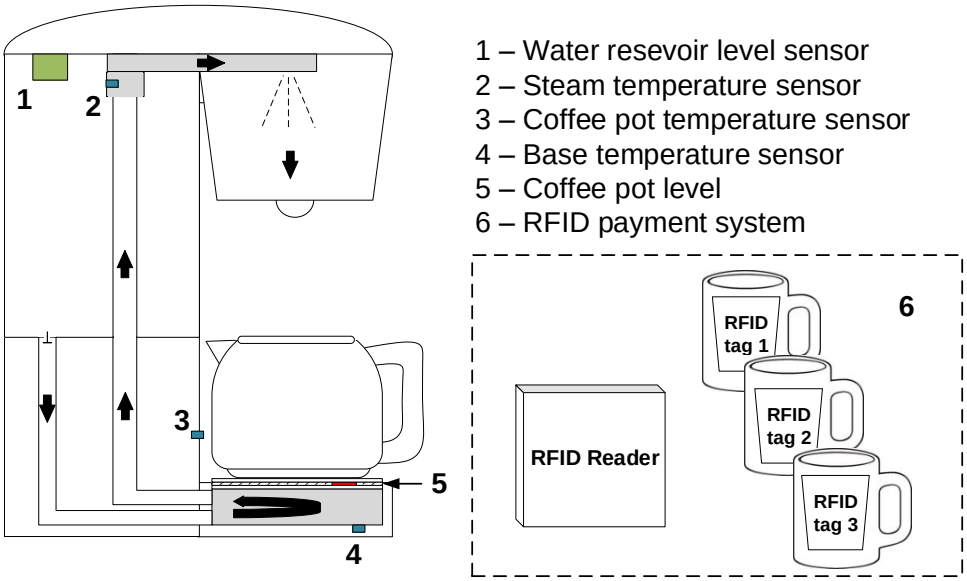


Figure 3.2: Placement of sensors on percolator with the RFID payment system feature.

Table 3.2: A Command defined in asset model.

Command	Parameters	Parameters Type
TriggerPulse	duty_cycle	integer

Table 3.3: Operating states defined in asset model.

Command	Value
Switched off	1
Idle	2
Brewing	3

3.1.2 Asset model

As mentioned in section 2.2.1 the asset model contains metrics such as data variables, commands, events, alarms and operating states. With the knowledge of the system objectives and how the SMART platform works the following variable in the asset model (Table 3.1) was defined in order to create a M2M application for monitoring a percolator remotely. In this project only the 'TriggerPulse' command is used for alternating the duty cycle of the TRIAC switching circuit and three operating states are declared. The operating states in Table **??** refers to the state of the percolator and not the system as a whole, because the 'monitoring' part of the system should always be working.

3.1.3 Flow of data

The system is designed to ensure that the various sensors are decoupled and does not strongly dependent on each other. Each sensor has its own voltage supply and ground wire in the event that if a component fails, it does not affect the entire system. If the TRIAC circuit fails it would affect the main mechanism if the system. Therefore it is vital to keep the TRIAC switching circuit separate from the rest. In Figure H.1 the percolator is also considered to be part of the flow of data. Although its data is not in digital format, it can be converted using the various sensors. In this figure it is clear that the Arduino has all the control because every data variable has to pass the Arduino in some way.

Chapter 4

Detail design

4.1 Hardware

4.1.1 Arduino

The microprocessor is the most important component of the project, because it determines how the rest of the system will interact with each other. Aspect which should be considered when choosing the microprocessor for a monitoring system is the following:

- The resolution of the ADC ports — This will determine the maximum accuracy of the measurements receive via the sensors.
- The amount of timers available — This is essential when certain tasks has to be conducted by the microprocessor independent of each other.
- The maximum allowable power consumption — This is neccessary to keep in mind, because most sensors and actuators requires significant of power to operate.
- The clock frequency of the Central Processing Unit (CPU) — This frequency sets the limit of executing code and the maximum rate for interrupts to occur (i.e. ADC sampling frequency and timer frequency)
- The amount of programmable memory required — This is crucial especially when dealing with AT commands that are solely based on ASCII characters.

For this project the Arduino Mega 2560 was chosen, because it has 16 analogue inputs. Analogue inputs are vital for measuring the output of most sensors available on the market. Using additional ADC IC packages would increase the software overhead and slow down the performance of the main program state machine. The Arduino Mega also has 54 GPIO pins (of which 15 can be used as PWM outputs). It also have 256 kilobytes which will make it easier to communicate with the GSM modem using AT commands.

The Arduino Mega 2560 board has an on-board programmer with an USB-interface that is connected to an ATmega16U2-chip which provides a virtual com port on the computer for the Arduino to communicate with any computer via UART. Additional features found in [25] that made this Arduino an attractive option to be used for this project is shown in Table D.1.

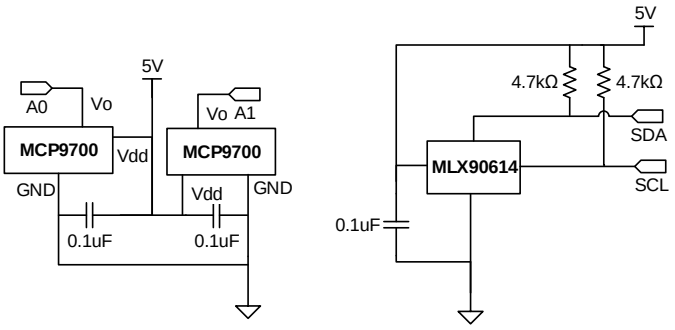


Figure 4.1: The interface circuits of the temperature sensors with the Arduino.

4.1.2 Temperature sensors

Base and steam temperature

Measuring the base and steam temperature the MCP9700 linear active thermistor is used instead of the passive thermistor. The MCP9700 was chosen, because it is inexpensive and had minimal hardware and software overhead than passive thermistor, which first needs to calculate Eq.(2.1). The MCP9700 outputs an analog value, which can be converted, using an ADC, to a temperature by only subtracting it with an offset value in software. Although the MCP9700 has many advantages it has one disadvantage that is its measurement accuracy varies with ±2°C.

Coffee pot temperature

In this project we used the Melexis MLX90614E-BAA infrared thermometer. Although this sensor costed R 349.95 there were no other alternatives that could measure the coffee pot's temperature. Due to its size of TO-39 package it did not effect the overall aesthetic value of the percolator. It is also suitable for this project, because it can measure temperatures between -40°C to 120°C, which is in the range of the temperatures we expect to measure from a percolator.

It can operate with a supply voltage (V_{DD}) between 2.7 and 5.5 volts. Therefore V_{DD} was chosen to be 5 volts, because the Arduino's I^2C-bus also operates with 5 volts.

With an open-drain configuration, the voltage is always high. The line should be "pulled down" using a switching transistor to transmit data. For the devices using I^2C, it is important to ensure that their voltage levels are compatible with each other. In Table 4.1 the voltage levels are shown for both devices.

When the I^2C-bus line is pulled low, the overall voltage must be a maximum of 0.8 V for both devices to interpret it as a logical zero ('0'), while the minimum value for representing a logical ('1') would be 2.1 V.

The total bus capacitance on both the SDA and SCL-line with the MLX90614 sensor connected is 20 pF. According to the Arduino's datasheet [25] the minimum and maximum internal pull-up resistor values is 1.53 kΩ and 30 kΩ respectively. Due to the uncertainty of this value for a pull-up resistor, another external pull-up resistor has been added in parallel

Table 4.1: Specifications taken into consideration for interfacing the Arduino and the Melexis MLX90614 infrared thermometer.

	$V_{IH(min)}$	$V_{IL(max)}$	$V_{OH(min)}$	$V_{OL(max)}$	$I_{IH(min)}$	$I_{OL(max)}$	C_{bus}
Arduino	3.5 V	1.5 V	5 V	0.4 V	3 mA	10 μA	10 pF
MLX90614	2.1 V	0.8 V	5 V	0.4 V	350 μA	3 mA	10 pF

to limit this value to change within an acceptable range. The pull-up resistor's value must be limited, in order for the voltage levels' to rise and o fall within a minimum duration of 20 ns [25]. By experimenting with different resistor values between 2.2 kΩ and 10k kΩ, it was found that 4.7 kΩ is a good value to use as a pull-up resistor. This resistor value is verified by calculating the time constant for the bus capacitance to charge and discharge using the formula, $t_{CAP} = R_{pu} \times 20pF$. R_{pu} is the two resistors in parallel's value. In this setup R_{pu} ranges between 1.15 kΩ and 4.06 kΩ, which means that t_{CAP} ranges between 23 ns and 81.2 ns. The range of t_{CAP} is acceptable given that the SMBus clock frequency is 100 kHz. If more devices are added to the I^2C-bus it will mean that the total bus capacitance should be re-calculated and the pull-up resistor should be changed accordingly to compensate for the timing requirements.

4.1.3 Water level measurement

It is required that the water reservoir water level and the coffeepot's coffee level should be measured. Each level is measured in a different manner. After considering the cost and taking the percolator's design into account, the two levels are measured as follows: the water reservoir's level is measured by using ultrasonic transducers, while the coffee pot's coffee level is measured by using an force-sensing resistor (FSR).

Water reservoir

The water reservoir could have been measured using the eTape liquid level sensor, but it costs R 635.00 compared to the ultrasonic transducers with their circuitry which costs R 30.00.

The reservoir's water level is measured by using two ultrasonic transducers as shown in Figure 4.2. This idea is originally obtained from [27], but it is adapted for this project. In [27] an H-bridge is used for amplification, but because the maximum distance that the signal can travel within the reservoir is only 16 cm, amplifying the signal is unnecessary. Therefore the transmitter is directly connected between the Arduino pin 11 and ground. An 40 kHz PWM signal is used for sending a pulse to measure its delay between the transducers. This delay is then processed to calculate the water level. The Arduino's ADC port cannot measure high frequency signals or signals with negative voltages, therefore the reflected signal required pre-processing. The desired result that this pre-processing should give is illustrated in Figure 4.3.

Figure I.3 shows the circuit design for converting the reflected signal to be compatible with the ADC port. The Arduino has only a positive 5 volts supply, therefore only single

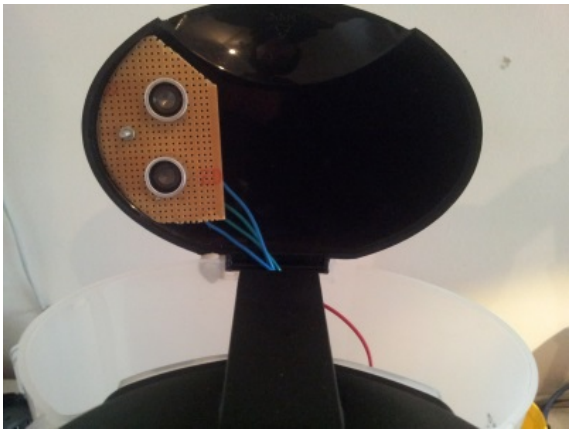


Figure 4.2: The placement of the ultrasonic transducers on the coffee percolator.

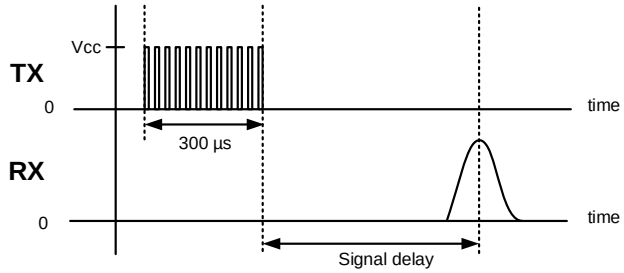


Figure 4.3: Configuration for using the ultrasonic transducers to measure the water level [26].

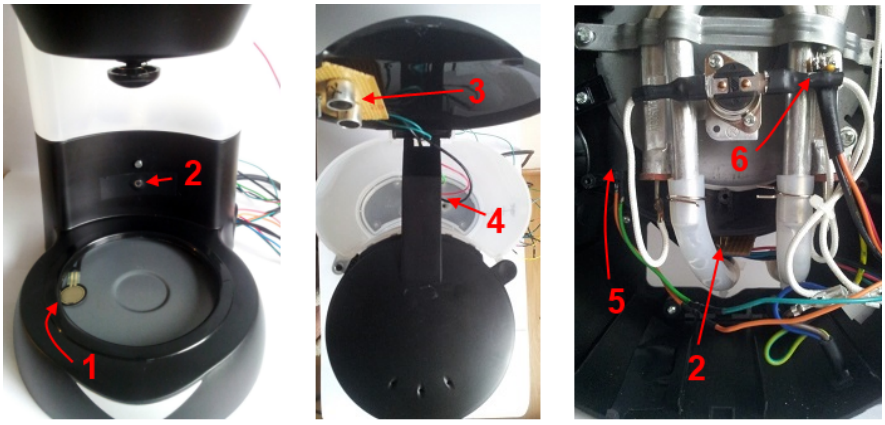
supply operational amplifiers (or op-amps) can be used. In this circuit the Texas Instruments TL07x op-amp is used and design equations from [28] is used to design these filters. The reflected signal first passes a band-pass filter centered around 40 kHz and then it passes a low-pass filter, with a cut-off frequency also at 40 kHz, to suppress the high frequency components that the band-pass has not suppressed adequately. After the low-pass filter the signal is amplified with a factor of 23. From the test data in section L.1, the smallest signal has a peak of 54 millivolts, which means if it is amplified then the signal has a peak of 1.242 volts. This amplified signal is passed through a peak detector, which gives the result illustrated in Figure 4.3. The largest peak voltage that the receiver will encounter, according to the test data, is 114 millivolts. When this signal is amplified it has a peak voltage of 2.62, which fits the voltage range of the ADC port.

In the software 21 ADC samples are stored in an array after the 40kHz pulse has been sent. In theory the distance is measured by calculating the time delay of the signal using a timer on the Arduino. But due to the short distance in which the signal travels, a cheaper solution is used. Instead of calculating the delay, the the highest value and its array index value is calculated rather and stored in a separate variable. The index value of where the highest value occur is directly proportional to the distance between the water level and the transducers.

Coffeepot water level

For measuring the coffee level within the coffeepot, the weight of the coffeepot is measured using a FSR. The water level can be calculated percentage-wise by measuring the weight of the coffeepot when it is full.

When the coffeepot is full it weights 1470 grams and when it is empty it weights 310 grams. The corresponding resistance values measured from the FSR is 3.6 kΩ when the coffeepot is full and 9.6 kΩ when it is empty. Due to time constraints, the FSR interfacing circuit is not designed yet.



(a) Front view (b) Top view (c) Bottom view

Figure 4.4: The Safeway coffee percolator.

4.1.4 Coffee percolator

The Safeway SCM-202 coffee percolator from Clicks was used, because it is inexpensive, easy to disassemble and had the same basic functions as the percolator discussed in section 2.1. Figure 4.4 will show where each sensor is placed to fulfil its particular function.

Figure I.4 shows the basic circuitry within the percolator. This circuit is connected with the TRIAC trigger circuit to switch the power on or off. The thermostat is to keep a maximum temperature of 80 - 85°C. This is just for safety for if the percolator exceeds 85°C and the sensors fail to detect it, the thermostat will prevent this from happening. It is unknown whether the glass coffee pot can endure temperatures above 85°C.

4.1.5 TRIAC driver circuit

The TRIAC's advantage over conventional mechanical switches is that it has a positive "on" and a zero current "off" characteristic, which means it does not suffer from the contact bounce or arcing inherent in electromechanical relays. Low energy consumption is required for triggering an TRIAC. Because of the low energy required for triggering a TRIAC, the control circuit can use inexpensive solid-state devices such as transistors, optically coupled drivers and IC's. Using the MOC3061 opto-coupled TRIAC greatly simplifies the control of the main TRIAC. Using built-in zero-crossing circuit in conjunction with the driver TRIAC to trigger the main TRIAC minimises the problem with false triggering and the problem with EMI being induced when the power supply is not switched at zero voltage.

Although the main TRIAC controls power of a resistive load it is still a good practice to use a snubber circuit across the main TRIAC to limit the rate of rise of voltage (dv/dt) across the mounting terminals to a value below the maximum allowable rating. This snubber circuit does not only limit the voltage rise during commutation, but also suppresses transient voltages that may occur as a result of disturbances of other electronic appliances. [21]

In Figure 4.5 the main TRIAC is effectively switched using only a 2N2222 transistor connected to the I/O port labelled as D2 on the Arduino. This 2N2222 switching circuit is designed to drive the MOC3061's light emitting diode(LED) with 15mA when 5V is being

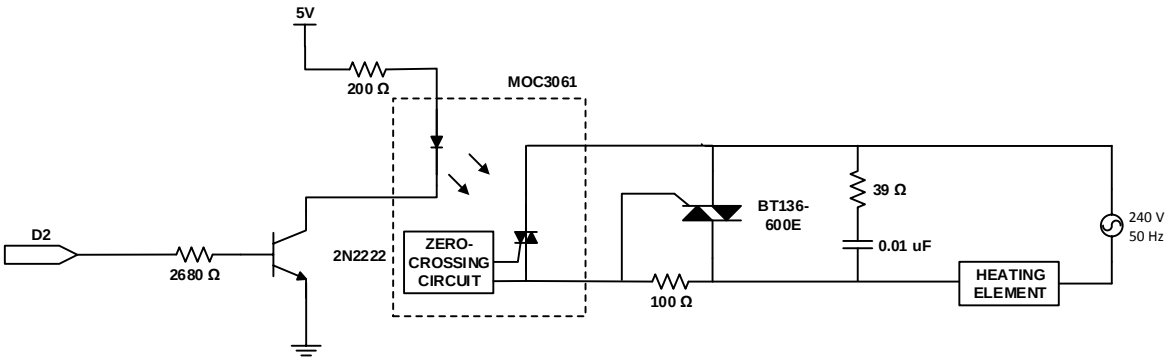


Figure 4.5: Interfacing circuit where the opto-coupled TRIAC is used to drive the main TRIAC.

applied at D2. The MOC3061 will ensure that the BT136-600E TRIAC will only be conducting when the MOC3061 LED is on and its zero-crossing circuit detects that the AC voltage has passed zero. When the BT136 is conducting then AC power will be dissipated by the heating element. Due to the built-in zero-crossing circuit only the cycle controlling method, as discussed in section 2.6.2, can be used for controlling the power being dissipated by the heating element.

Control signal

Using the 16-bit Timer 3 configured to generate an interrupt every second of the Arduino it made it possible to control the power independently of the main program loop. The interrupt routine was programmed to accept a parameter which specifies the duty cycle. This duty cycle effectively defined the duration for the LED within the MOC3061 to be on and also to be off. For example, if the duty cycle was 10, then the LED would be on for 100 ms and 900 ms off. If the duty cycle is set to 100, then the LED will be constantly conducting until the duty cycle is decreased.

Power considerations

The power being dissipated within the BTA136-600E TRIAC is calculated with Eq. (4.1) supplied by [29].

$$P_{TRIAC} = V_o \times I_{T(AVE)} + R_s \times I_{T(RMS)} \tag{4.1}$$

The TRIAC knee voltage (V_o) and the TRIAC's slope resistance (R_s) is obtained from [30]. $I_{T(RMS)}$ is derived from dividing the 900 watt rated heating element with the root mean squared (RMS) mains supply voltage of 240 V, which leads to 3.75 A. This result is used for calculating $I_{T(AVE)}$, which equals to 3.38 A. Then P_{TRIAC} would to be 4.629 watt.

P_{TRIAC} is significantly high for a TO-220 package, therefore one must check whether this TRIAC needs a heat-sink for dissipating the heat when operating. From the TRIAC's

Table 4.2: Summary of component's voltage and current requirements to be used for designing the DC power supply with.

Component	**Voltage**			**Current**		**Reference**
	Min	Max	Typ	Max	Typ	
FXT009 GSM Modem	4.75 V	32 V	12 V	2 A	600 mA	[7]
Arduino Mega 2560	6 V	20 V	9 V	1 A	200 mA	[24], [31]
2 × MCP9700	2.3 V	5.5 V	5 V	24 mA	12 μA	[11]
MLX90614	2.7 V	5.5 V	3.3 V	2 mA	1 mA	[13]
TL074C (3 Op-amps)	-18 V	18 V	*	7.5 mA	4.2 mA	[32]
TRIAC circuit	0 V	12 V	5 V	20 mA	15 mA	
MAX232 IC	4.5 V	5.5 V	5 V	10 mA	8 mA	
LM324 Op-amp	-16 V	32 V	5 V	1.2 mA	0.7 mA	
Design requirements	-18 V	32 V	12 V	3 A	830 mA	

datasheet [30] the TRIAC has a maximum junction temperature ($T_{j(max)}$) of 125 °C. When no heat-sink is used the thermal resistance between the temperature of the TRIAC's junction (T_j) and the ambient temperature (T_{amb}) is typically 60 °C/K — more formally stated as $\Theta_{j-amb} = 60$ °C/K. This means, for instance, if T_{amb} is 10°Cthen T_j will reach a maximum temperature of 287.74 °C. This is unacceptable, therefore a heat-sink is mandatory. The rectangular aluminium heat-sink used in this project to cool the TRIAC has dramatically reduced the thermal resistance, Θ_{j-amb}, to 4.6 °C/K. Now the maximum allowable ambient temperature for the TRIAC to operate in is increased to approximately 100°C.

4.1.6 DC power supply

The GSM modem and the Arduino are connected in parallel with the DC power supply. For building the power supply the schematic in Figure I.2 is used. The power supply has two separate loads in parallel, which are the GSM modem and the Arduino. The Arduino's current consumption is limited to 1 A by the NCP1117 voltage regulator connected to the power jack on the board.

The typical values in Table 4.2 are used to design the DC power supply. That means the power supply would output a constant 12 V, which can deliver a maximum of 1 A to the Arduino and the GSM modem separately. A 10VA printed circuit board(PCB) transformer from Block is used to transform the 240 V RMS mains to 18 V RMS. This transformer can deliver a maximum current of 1.8 A, which limits the GSM modem and the Arduino from an excess of current. Although a maximum current of 3 amperes is calculate in Table 4.2 it is highly unlikely, because only the GSM modem would maybe require more than 1 amperes if it struggles to get good reception and the Arduino is limited to 1 amperes. For the design an inexpensive diode bridge component is used, because it is elegant and allows a maximum of 2 amperes to flow through it. For the design a ripple voltage of 5 volts is specified to

reduce the capacitance of the ripple capacitor, which is 8.8 mF. The two LM7812C voltage regulators from Fairchild is placed in parallel to created a regulated 12 volts for the GSM modem and the Arduino to use.

Due to time constraints the DC power supply did not work properly, therefore all the components listed in Table 4.2, except the GSM modem, used the Arduino's power supply. This setup is not desirable, because all the current flows through the surface mounted NCP1117 voltage regulator which has no separate heatsink and therefore could damage your Arduino board due to excessive heat.

4.1.7 Current sensor

The maximum current to be used by the 900 W heating element was calculated to be 3.75 A RMS. A linear current transformer was used, instead of an hall effect sensor, because it costs R 100.00 on average for a sensor to measure a maximum of 3.75 A RMS. The Talema AX-1000 current transformer cost R 36.23.

In [33] states that by using a 200 Ω resistor in parallel with the sensor terminals, will give a linear transfer function between input AC current and output voltage, which output ranges from 0 to 1 V with a input that ranges from 0.1 A RMS to 5 A RMS. A test screenshot of the current sensor's output can be seen in section L.2. The development of the interface circuit has not been done due to time restriction, but it would be very similar to the voltage sensor's interface circuit.

4.1.8 Voltage sensor

The voltage sensor used in this project is an ordinary voltage transformer with a voltage biasing circuit to interface it with the Arduino. The specific part used for measuring voltage was obtained from electronic components that were about to be recycled. It had minimal documentation, but it still operated as expected from an ordinary voltage transformer. Initially it was thought to design a circuit that converted a down-scaled AC signal to a DC voltage using a low-pass filter. This design seemed to work on paper, but when it was implemented it did not deliver any significant results. Then the circuit design suggested by [34] was followed. Figure 4.6 shows a slightly adapted version. This circuit solved the problem by calculating the RMS voltage using an algorithm.

From Figure 4.7 the peak voltage measured at the output terminals of the voltage transformer is 16 volts. This peak voltage should be scaled down and an offset voltage should be added to fit the measurement within the Arduino's ADC port which ranges from 0 - 5 volts. The scaling is done with a voltage divider circuit and a offset is added by using two large 470kΩ resistors to minimise current consumption. The resistors divides the 16 voltage with a factor of 11. Thus the maximum voltage to appear at the ADC port, labelled as A9, is 3.63 volts assuming that the offset voltage is 2.5 volts (or $V_{cc}/2$), while the minimum voltage to appear at A9 is 1.13 volts. The scaling also left some space for the worst case scenario where the supply voltage (V_{cc}) is 4.8 volts.

The additional Arduino classes pre-configures the 10-bit ADC module, embedded within the ATmega2560-chip, with an effective ADC sampling frequency of approximately 8900 Hz. This is fast enough for sampling the 50 Hz mains voltage, because the minimal frequency to

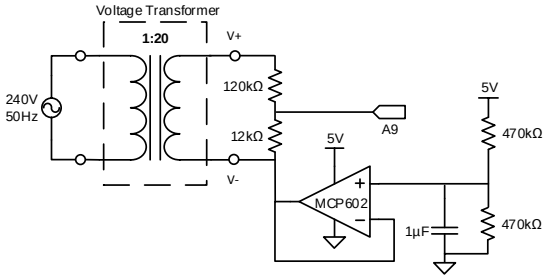


Figure 4.6: Interfacing circuit for the Arduino where a voltage transformer is used to measure the mains voltage connected to the percolator.

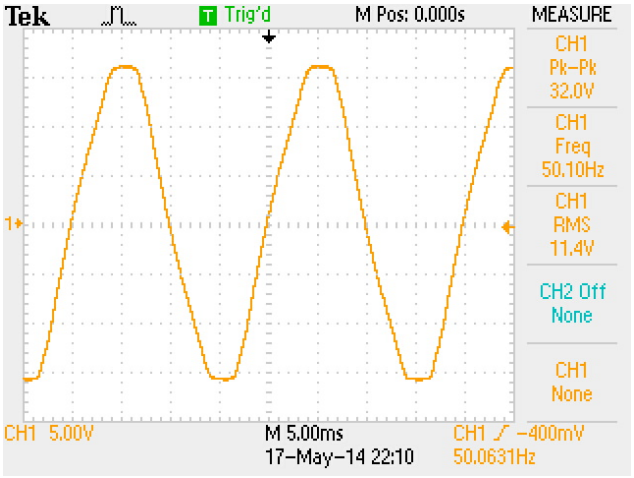


Figure 4.7: Output sinusoidal waveform of the voltage transformer with mains voltage as input.

avoid aliasing would be 100 Hz minimum. For sampling a 50 Hz signal at 8900 Hz one needs at least 178 samples to sample one period.

4.1.9 RFID payment system

This feature is design only to accommodate three users, because the SMART platform does not allow the user to create additional variables. If additional variables have to be created then the asset model must be adjust to accommodate more users. Although this feature has been implemented using the Electrically Erasable Programmable Read-Only Memory (EEPROM) embedded in the Arduino, the program flow diagrams are not included in this report due to time constraints. In Appendix G the source code for this feature is kept in a git repository with the file called "rfid_test.ino". The Adafruit NFC breakout board were used as the RFID-reader in this project.

4.2 Details of the software

The software for this system consists out of one main program, which loops continuously, and a few subroutine functions that are called via interrupts being triggered. Figure E.1 shows the overall flow control of the main program. Interrupts, that are set, are mainly used for controlling the percolator's power duty cycle by using the Arduino's Timer 3 interrupt subroutine function. Using interrupts reduces the processing load on the main program loop and encourages the system to execute unrelated tasks concurrently that are crucial for monitoring the percolator's current state.

The main program starts to monitor the percolator after executing the set-up method successfully. The first step in this loop is to call "serial1_event"-method that stores every byte received from UART1 into a string called "inputString". In Figure E.2 shows that the "serial1_event"-method loops until it receives the character '\n'. This character, sent from

the GSM modem to the Arduino, indicates the end of the message being transmitted to the UART1 serial port. The "stringComplete"-flag is set to indicate that there are new data available to be processed. It is expected to receive either "OK" or "ERROR" or an AWTDA-formatted response. Therefore the minimum of characters to appear in "inputString" should be more than two for the program to process it. If the "inputString" is longer or equal to two characters, including the character '\n', then first seven characters are extracted from "inputString" into a temporary string. This temporary string is then evaluated to detect what type of response it is. If the response includes "+AWTDA:" then the "busy_wait"-flag is set and the response further analysed in the "unpack_response"-method. By setting the "busy_wait"-flag forces the Arduino to remain idle while the response from the SMART platform is being analysed.

In the event where the "stringComplete"-flag is not set, the program moves on to the next task, where it checks the status of the "busy_wait"-flag and the "sensor_updated"-flag before it continues to read and process the updated sensor values. The "sensor_updated"-flag is set within the Timer 4 interrupt subroutine method. The purpose of this flag is to ensure good control regulation between the interrupt subroutine method, which also reads the current ADC port values, and the main program. This interrupt subroutine is kept very short in order to minimise the impact it has on the main program's execution time. When both the "busy_wait"-flag and the "sensor_updated"-flag is cleared then UART1 transmits an AWTDA command to the SMART platform as follows:

```
Serial1.print(F("AT+AWTDA=d,\"Coffee.temp\",3,\"base,INT32,"));
Serial1.print(baseTemp);
Serial1.print(F("\",\"steam,INT32,"));
Serial1.print(steamTemp);
Serial1.print(F("\",\"coffee,INT32,"));
Serial1.print(coffeeTemp);
if(sendATcommand("\"", "OK", 2000) == 1)
{
Serial.println(F("Sensor data updated"));
}
```

This AWTDA command is specified to transmit three parameters, in this case the base temperature; the steam temperature and the coffee pot temperature. The "sendATcommand" is used to if this AWTDA command had failed or if it had respond with "OK". It is not placed within a 'while'-loop, but rather with a "if"-statement to encourage the main loop continue and try again in a later stage. This methodology is used in the event network coverage has been lost by the GSM modem. The "sendATCommand" can be given a maximum time for receiving the expected response response. In this case it is set at 2000 ms or 2 seconds. The program flow for the "sendATCommand"-method is shown on Figure E.3.

When the "inputString" starts with "+AWTDA:" and "busy_wait"-flag is not set, then the "unpack_response"-method is called. The flow diagram for this method is shown on Figure E.4. In the event that a command is received (i.e. the type value in "unpack_response" equals 'c') then the "process_command"-method is called to execute the command and to send an acknowledgement message to the platform.

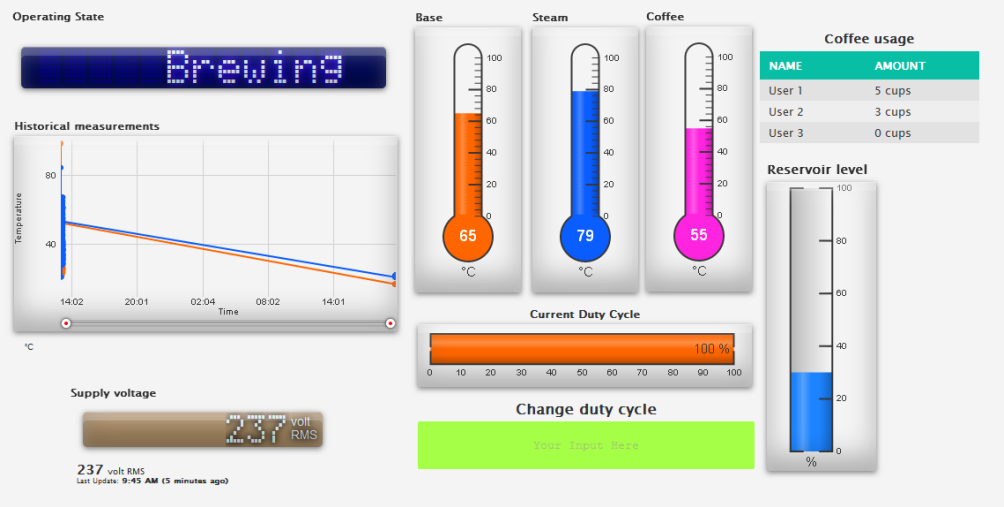


Figure 4.8: Screenshot of dashboard.

4.2.1 Dashboard overview

Figure 4.8 shows the end-users interface. The duty cycle of the percolator can be set here to control the power delivered to the heating element.

Chapter 5

Measurements and Tests

5.1 Testing of GSM modem communication

The objective of this test is to determine whether the Arduino and the GSM modem can communicate through the UART serial interface.

5.1.1 Hypothesis

The modem and the Arduino operates at different voltage levels. The MAX3323 IC is used to translate these voltage levels for these devices. Effectively it acts as an interface between the two serial interfaces by ensuring that the bits being sent and received are interpreted correctly.

5.1.2 Test design

This test is conducted with source code that only executes a routine that checks whether the Arduino can communicate with the GSM modem through UART. Two hardware serial ports on the Arduino are used. One for communicating with the GSM modem and the other one with the computer via the USB-interface. The terminal program is used for displaying the response from the GSM modem and also for debugging purposes. This test transmits three AT commands from the Arduino to the GSM modem. In Table 5.1 the specific commands are listed with the expected responses. See [8] for more information about the AT commands. The test is conducted as follows:

Table 5.1: Test AT commands sent to the GSM modem via the Arduino.

Command	**Expected response**	**Range**
AT	OK	N/A
AT+CREG	+CREG:0,1	N/A
AT+CSQ	+CSQ: <rssi>, <ber>	<rssi> : 11 - 31

1. Connect the GSM modem's serial port to the Arduino via the RS-232 converter (MAX232). Wait for the GSM modem's status LED to blink at a rate of 200 ms per second.
2. Both the Arduino and the GSM modem must be powered by the 12V DC power supply.
3. Load the test source code named, "serial_modem_test.ino" onto the Arduino. Refer to Appendix G for locating the code.
4. Open a terminal program (e.g. Serial monitor) and set the baud rate to 9600 (8-N-1).
5. Press the reset button on the Arduino to start up the test program.
6. Transmit the character '1' via the terminal to send "AT" to the modem and wait for response.
7. Transmit the character '2' via the terminal to send "AT+CREG" to the modem and wait for response.
8. Transmit the character '3' via the terminal to send "AT+CSQ" to the modem and wait for response.

If all three commands send the expected responses back then the connection is considered to work properly.

5.1.3 Results

The GSM modem's LED indicator blink at a rate of 200 ms per second. The Arduino's TX and RX LEDs flickered as the response was transmitted and received within a few milliseconds. The test program sent the following response:

```
*****   Send 'AT':   *****
Response: OK

*****   Send 'AT+CREG?'   *****
Response: +CREG: 0,1
Info: Registered on the home network

*****   Send 'AT+CSQ'   *****
Response: +CSQ: 15,0
Info: Sufficient signal strength
```

The data that the modem sent back is shown next to each "Response:".

5.2 Testing of the temperature sensors

The objective of this test is to determine whether the three temperature sensors (two MCP9700 IC's and the Melexis MLX90614 sensor) located on the coffee percolator is calibrated and able to track the temperature in the part of the percolator allocated to it accurately over a predefined time period.

5.2.1 Initial conditions

The following initial conditions are set for this test:

- The ambient temperature of measured to be 18°C using an external calibrated thermometer.
- No coffee is placed in the funnel during the test.
- The coffee percolator has been off for a few hours, before this test started.
- The heat switch is regulating the temperature of the heating element.

5.2.2 Hypothesis

When the percolator is switched on by the TRIAC trigger signal with a 100% duty cycle, the base temperature sensor's output will be the first to increase gradually, because it is situated the closest to the heating element. Secondly the steam temperature's output will also increase gradually, but it will have periodical "spikes" that indicates the hot water being pulsed along the sensor's die. The temperature that the infrared thermometer detects will remain constant at the beginning due to the slow transfer rate of water into the coffee pot. The temperature will gradually increase as the coffee level rises in the coffee pot. When the percolator is switched off, each sensor will gradually decrease in value at different rates. The values obtained from the sensors will be processed and will be sent periodically to the SMART platform by using the appropriate AWTDA command.

5.2.3 Test design

The test requires the two MCP9700 IC's to be connected to the pre-allocated pins, which is allocated onto the Arduino. The MLX90614 sensor is also required to be connected via the Arduino's I^2C-interface. Before the test is executed the instructions shown in section K.1 has to be conducted first:

The source code is available in a git repository listed in Appendix G.

5.2.4 Results

The results is different than predicted in the hypothesis. The base temperature was not the first temperature to rise, instead it was the steam temperature due to the water being heated very rapidly and pushed through to the shower head. From Figure 5.1 it is clear that the steam temperature showed some 'spike' values as the boiling water passed through the

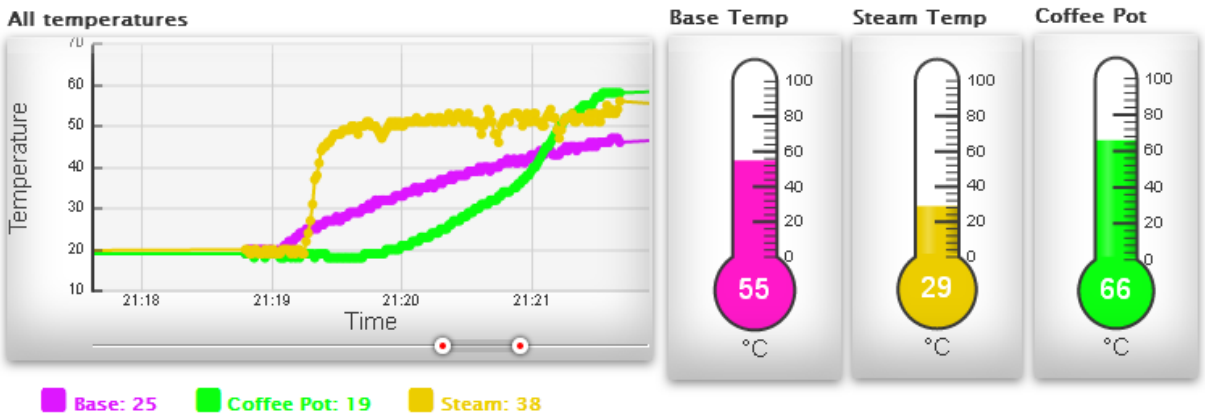


Figure 5.1: Screenshot of dashboard after the test has finished.

shower head. The coffee pot's temperature showed increase of temperature about a minute after the test was started.

5.3 Test of the ultrasonic sensor

The objective of this test is to determine whether the ultrasonic sensor can be successfully used to determine the water level within the water reservoir.

5.3.1 Hypothesis

Using the sonar principle, it is possible for these ultrasonic transducers to measure the water level by calculating the delay between the sent and received signal. During operation, if the water level decrease, the delayed signal will increase and will reach a maximum value when the reservoir is empty.

5.3.2 Test design

For conducting this test follow the instructions in section K.5.

5.3.3 Results

The results are good but does not show great resolution between the different water level. It is still useful for this system, because it shows that there is a linear relationship between the index value and the amount of water within the resevoir.

Table 5.2: Output of ultrasonic sensor versus amount of water in reservoir

Amount of water	**Maximum index**
None	11
465 ml	8
920 ml	4
Full	3 or 4

5.4 Test of voltage sensor to measure the mains voltage

The objective of this test is to determine whether ADC port, where the output of the voltage sensor is connected, can successfully obtain samples and translate them to the actual RMS voltage using the Arduino.

5.4.1 Hypothesis

When the mains voltage is applied to the primary side of the transformer, it will induce a scaled-down voltage on the secondary side, which can be sampled using the ADC port of the Arduino and then process these samples to calculate the RMS voltage.

5.4.2 Test design

This test is conducted with test code that samples the ADC port labelled as A9. These samples are stored within an array of integers that is further processed to obtain the actual RMS voltage. A multimeter which reads RMS AC voltages will be used to verify the results showed by the Arduino.

The test is conducted by following the instructions in section K.4.

The source code is available in a git repository listed in Appendix G.

5.4.3 Results

Table 5.3: Voltage sensor results

Mains supply	**Arduino RMS Voltage**	**Multimeter RMS**
On	237	240
Off	10	0

As seen in the table above the Arduino's measurement were not the same as the multimeter. This can be cause by noise on the ADC port or the offset value of the voltage sensor circuit. This problem can be eliminated by using a band-pass filter.

Table 5.4: List of RFID-cards' UID used in this test

Card No.	UID	Description
1	CA-2D-EA-92	Student card
2	2B-B6-EB-63	Booktag A
3	BB-52-F2-63	Booktag B
4	4D-4D-9E-F7	Blank Mifare card

Table 5.5: Initial conditions for RFID test

User No.	UID	Cup usage
1	FF-FF-FF-FF	0
2	FF-FF-FF-FF	0
3	FF-FF-FF-FF	0

5.5 Test of the RFID payment system

The objective of this test is to determine whether the Arduino can keep track of the amount of swipes each RFID-tag has made with the Adafruit PN532 breakout board.

5.5.1 Hypothesis

When the test program is started, with the RFID shield stacked upon the Arduino, the program will enter an idle state and wait for a RFID-card to interact with. If a RFID-card is held near to the Arduino, the test program will try and compare the new card's UID with all the other UID that already exists on the system. If the new card already exists then the system will increment the "coffee usage"-counter with one. If there are already exists three different UID's and another new card is held near to the Arduino, the system should not register the UID of the new card. After each RFID-card interaction the system synchronise the coffee usage with the SMART platform.

5.5.2 Test design

This test is conducted by using four different RFID-tags with their UID listed in Table 5.4

5.5.3 Initial conditions

The conditions at which this test is started is demonstrated using Table 5.5.

5.5.4 Test design

This test is conducted using the instructions listed in section K.2.

The source code is available in a git repository listed in Appendix G.

5.5.5 Results

The results that were obtained after executing the two tests are shown in section M.3:

The first four steps of Table K.1 and Table K.2 was to register the first three RFID-card to the system and ignore the fourth one. The expected result for the first test is that the

system will ignore card number 4, while in the second test card number one was ignored. In both tests the system acted as expected according to the test logs in Appendix N. Figure M.6 and M.7 shows the correct results for the first and second tests respectively.

See Appendix N for the terminal outputs for both tests.

5.6 Test of the TRIAC control circuit connected to the percolator

The objective of this test is to determine whether the TRIAC control circuit switches the supplied mains voltage correctly. During this test the duty cycle of the TRIAC trigger signal will be tested at duty cycles of 10%, 30%, 50% and 70%.

5.6.1 Hypothesis

When the percolator is switched on with a 0% duty cycle TRIAC trigger signal, then the coffee percolator should remain switched off. It is expected that when the TRIAC trigger signal is being switched with a duty cycle that mains voltage will be applied accordingly.

5.6.2 Test design

This test is conducted with source code that accepts the duty cycle as input to set up the waveform of Timer 3 on the Arduino.

The test requires an oscilloscope to verify that the measurements are correct. The oscilloscope should be able to measure the output waveform of Timer 3 on pin D2. The current sensor is used to measure the voltage waveform applied to the heating element.

The test is conducted by following the steps listed in section K.3

The source code is available in a git repository listed in Appendix G.

5.6.3 Results

See Appendix M for additional oscilloscope screenshots to interpret the results. In all four figures the yellow square wave, or CH1, represents Timer 3 output waveform, while the blue waveform, or CH2, represents the output of the current sensor. In all four figures one can clearly see that whenever the signal on CH1 goes high, it supplies AC power to the heating element of the percolator. Therefore the power being supplied to the percolator is successfully controlled with the TRIAC control circuit using the cycle control method.

Chapter 6

Conclusions

6.1 Objectives achieved

The proposed system was designed successfully, but not implemented entirely. Thus not all the objectives listed in section 1.3 has been reached yet.

Sensors that were successfully implemented were the three temperature sensors, which are evident from the tests. The ultrasonic sensor was successfully designed and implemented. Although it did not have a good measurement resolution, it could measure the reservoir water level in a qualitative manner. For example, the reservoir's water level was indicated as 100%, 50% or 30% full. The voltage sensor was implemented with partial success. The voltage transformer saturated before it reaches 240 V RMS, which affected the measurements taken in section 5.4.3, which differs from the expected values. The RFID payment system is implemented successfully, which is evident from the test carried out. Only problem with the RFID payment system's implementation is that it is centered on the EEPROM, which is problematic because this type of memory does not last very long. The EEPROM was chosen as a last resort solution, because it was not possible to retrieve any data from the SMART platform using the "AWTDA=dh" command. The TRIAC has been successful with controlling the power delivered to the percolator. However the cycle control method causes the voltage sensor to given incorrect measurements when the duty cycle of the TRIAC switching circuit is varied between 0 and 100 percent. A monitoring program that could retrieve and analyse AWTDA unsolicited responses with ease and could also execute the command "TriggerPulse", which allowed the end user to specify the duty cycle by sending the "TriggerPulse" command from the SMART platform.

The level of the coffee pot could also not be implemented due to time constraints. The current sensor could not be tested, because it was damaged before being tested.

The DC power supply was designed, but not built due to time constraints.

Effectively the Arduino Mega could measure the temperature and measure the reservoir's water level while controlling the percolator by using the TRIAC circuit, with the assistance Timer 3 to specify the duty cycle that determine the amount of power being delivered to the percolator.

6.2 Future work

For future work one can use more than one sensor to determine the base temperature. They can be scattered into an array, which also can increase the accuracy of the sensors. One could add a water pump that pumps water into the reservoir when it is empty. Replacing the DC power supply with a switch mode power supply would not be a bad idea if that would reduce the power consumption. The AWTDA command "AT+RTC?" can be used with the real-time clock module within the Arduino to keep track of real time. This will come in handy when implementing a logging system. There is still not an effective way found for measuring the coffeepot's water level with the FSR. If an FSR has been used it would not last, because it cannot operate at temperatures ranging between 80°C and 85°C. The power consumption measurements can be much improved using the Cirrus Logic Two Channel Energy Measurement IC (CS5490) had it been known to exist earlier during the project.

References

[1] M. Brain and J. Toothman. (2014) HowStuffWorks "How Coffee Makers Work". [Online]. Available: http://home.howstuffworks.com/coffee-maker.htm

[2] Coffee Research Institute. (2014) Coffee Brewing — CoffeeResearch.org. [Online]. Available: http://www.coffeeresearch.org/coffee/brewing.htm

[3] SeabeeCook.com. (2014) Brewing Coffee the Navy Way. [Online]. Available: http://www.seabeecook.com/cookery/recipes/navy_coffee.htm

[4] Sierra Wireless. (2014) Sierra Wireless - Add cellular to apps connectivity with programmable modems. [Online]. Available: http://www.sierrawireless.com/productsandservices/AirLink_Gateways_Modems/Programmable_modems.aspx

[5] Trinity Telecommunication. (2014) Trinity SMART Sense — SMART View Series. [Online]. Available: http://www.trintel.co.za/site/trinity/documents/Trinity_SMART_Sight_Brochure_July_2010.pdf

[6] Sierra Wireless. (2014) Sierra Wireless Airlink Xtend and GL Series. [Online]. Available: http://www.trintel.co.za/site/trinity/documents/SW_FXL_Series_datasheet.pdf

[7] *User Guide Airlink FX Series*, Sierra Wireless, 13811 Wireless Way, Richmond, BC, Canada, 2013, revision 8.2.

[8] *AT Commands Interface Guide for Firmware 7.52*, Sierra Wireless, 13811 Wireless Way, Richmond, BC, Canada, 2013, revision 6.0.

[9] Wikipedia. (2014) Thermistor — Wikipedia the Free Encyclopedia. [Online]. Available: http://en.wikipedia.org/wiki/Thermsistor

[10] J. S. Steinhart and S. R. Hart, "Calibration curves for thermistors," *Deep Sea Research and Oceanographic Abstracts*, vol. 15, no. 4, pp. 497–503, 1968.

[11] "Low-Power Linear Active Thermistor ICs (MCP9700)," Microchip, USA. [Online]. Available: http://ww1.microchip.com/downloads/en/DeviceDoc/21942e.pdf

[12] "MLX90614 family Single and Dual Zone Infrared Thermometer in TO-39," Melexis — Microelectronic Integrated Systems, 2009.

[13] Melexis, "Application Note — SMBus communication with MLX90614," 2008.

[14] "Continuous Fluid Level Sensor (PN-12110215TC)," Milone Technologies, Inc., 17 Ravenswood Way, Sewell, New Jersey, 08080. [Online]. Available: http://www.milonetech.com/uploads/Std_eTape_Datasheet.pdf

[15] Wikipedia. (2014) Ultrasound — Wikipedia the Free Encyclopedia. [Online]. Available: http://en.wikipedia.org/wiki/Ultrasound

[16] ——. (2014) Force-sensing resistor — Wikipedia The Free Encyclopedia. [Online]. Available: http://en.wikipedia.org/wiki/Force-Sensing_Resistor

[17] ——. (2014) Current Sensor — Wikipedia the Free Encyclopedia. [Online]. Available: http://en.wikipedia.org/wiki/Current_sensor

[18] B. Guru and H. Hiziolu, *Electromagnetic Field Theory Fundamentals*. The Edinburgh Building, Cambridge CB2 8RU, UK: Cambrigde University Press, 2004.

[19] ElectronicTutorials. (2014) Current Transformer Basics and Current Transformer Theory. [Online]. Available: http://www.electronics-tutorials.ws/transformer/current-transformer.html

[20] ElectronicsTutorials. (2014) Electrical Relay. [Online]. Available: http://www.electronics-tutorials.ws/io/io_5.html

[21] ON Semiconductor. (2006, Nov) Thyristor Theory and Design Considerations. [Online]. Available: http://www.onsemi.com/pub_link/Collateral/HBD855-D.pdf

[22] Wikipedia. (2014) Radio-frequency identification — Wikipedia the Free Encyclopedia. [Online]. Available: http://en.wikipedia.org/wiki/Radio-frequency_identification

[23] Atmel Corporation, "Application note — understanding the requirements of iso/iec 14443 for type b proximity contactless identification cards," 2005. [Online]. Available: http://www.atmel.com/Images/doc2056.pdf

[24] Arduino. (2014) Arduino — Arduino - ArduinoBoardMega2560. [Online]. Available: http://arduino.cc/en/Main/ArduinoBoardMega2560

[25] (2014) 8-bit Atmel Microcontroller with 16/32/64KB In-System Programmable Flash. Atmel Corporation. 1600 Technology Drive, San Jose, CA 95110 USA. [Online]. Available: http://www.atmel.com/Images/Atmel-2549-8-bit-AVR-Microcontroller-ATmega640-1280-1281-2560-2561_datasheet.pdf

[26] Wikipedia. (2014) Ultrasonic Level Measurement — AutomationWiki. [Online]. Available: http://automationwiki.com/index.php?title=Ultrasonic_Level_Measurement

[27] K. Wong. (2014) Kerry D. Wong — Blog Archive — A Sensitive DIY Ultrasonic Range Sensor. [Online]. Available: http://www.kerrywong.com/2011/01/22/a-sensitive-diy-ultrasonic-range-sensor/

[28] Texas Instruments, "Application Report — Filter Design in Thirty Seconds," Dec 2001. [Online]. Available: http://www.ti.com/lit/an/sloa093/sloa093.pdf

[29] P. Semiconductors, "Philips Semiconductors AN10384 — triacs: How to calculate power and predict tj(max)," 2005.

[30] "Triacs sensitive gate — bta136 e series," Philips Semiconductors, 2001.

[31] "NCP1117 — 1.0 A Low-Dropout Positive Fixed and Adjustable Voltage Regulators," ON Semiconductors, P.O. Box 5163, Denver, Colorado 80217 USA, 2009. [Online]. Available: http://www.onsemi.com/pub_link/Collateral/NCP1117-D.PDF

[32] (2014) TL07x Low-Noise JFET-Input Operational Amplifiers. Texas Instruments. Post Office Box 655303, Dallas, Texas 75265. [Online]. Available: http://www.ti.com/lit/ds/symlink/tl074.pdf

[33] "AX Series 50/60Hz Current Transformer," Talema, 2001. [Online]. Available: http://docs-europe.electrocomponents.com/webdocs/11b0/0900766b811b0377.pdf

[34] OpenEnergyMonitor. (2014) Measuring AC Voltage with an AC to AC power adapter. [Online]. Available: http://openenergymonitor.org/emon/buildingblocks/measuring-voltage-with-an-acac-power-adapter

Appendices

Appendix A

Project planning schedule

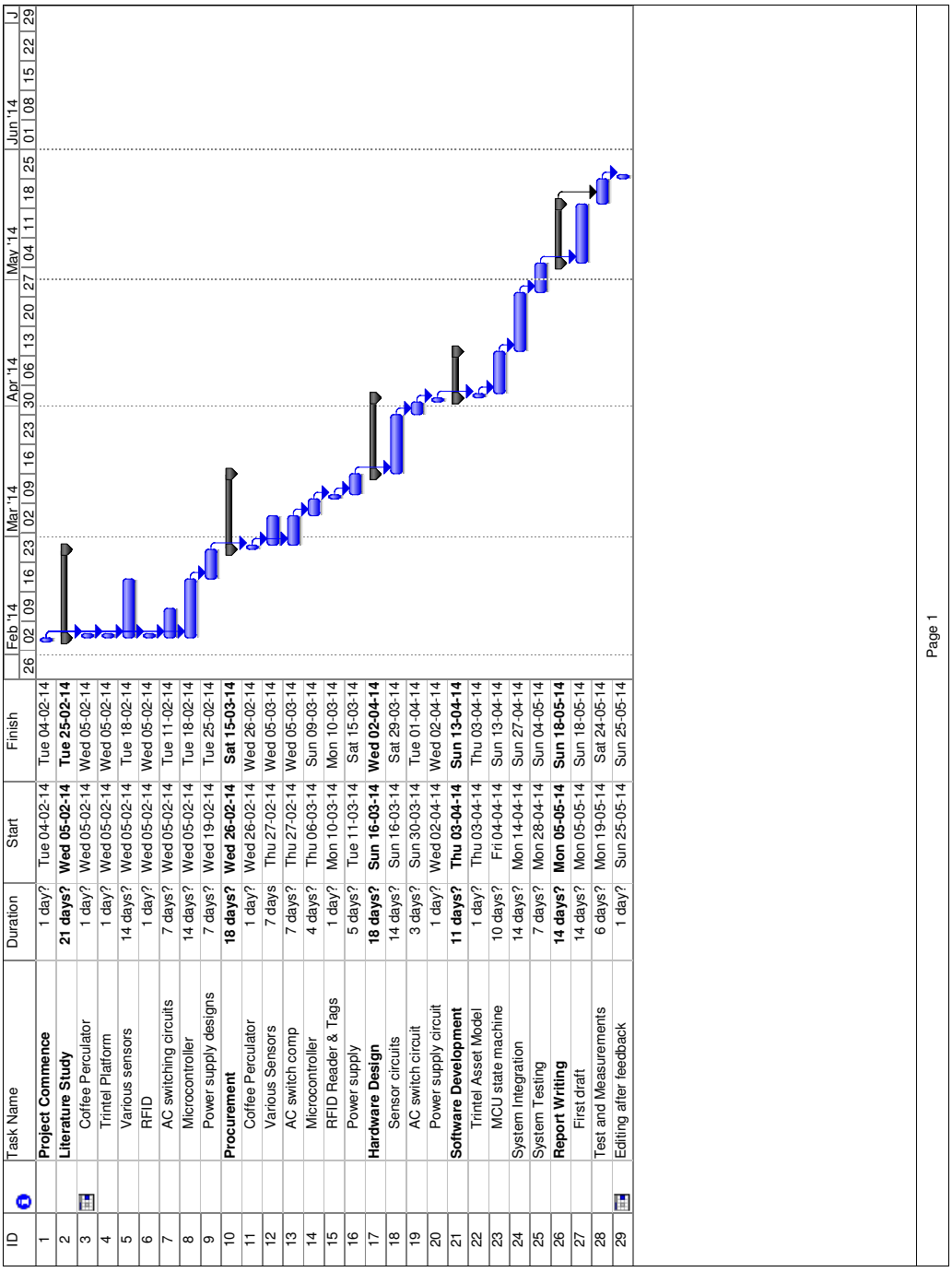


Figure A.1: Project planning schedule using a Gantt chart.

Appendix B

Project specification

Develop a "connected coffee machine" by integrating it with sensors (temperature, water levels and RFID) for measuring and actuators (controlling power supply) via the SMART platform's interface. Also develop user friendly graphical interface using the platform for monitoring the system.

Appendix C

Outcomes compliance

Table C.1: ECSA outcomes criteria.

Outcome	**Reference**	
	Sections	Pages
1. Problem solving: Demonstrate competence to identify, assess, formulate and solve convergent and divergent engineering problems creatively and innovatively.	3	20 - 22
2. Application of Scientific and Engineering Knowledge: Demonstrate competence to apply knowledge of mathematics, basic science and engineering sciences from first principles to solve engineering problems.	4	23 - 33
3. Engineering Design: Demonstrate competence to perform creative procedural and non-procedural design and synthesis of components, works, products and processes.	3 - 4	20 - 33
4. Investigations, experiments and data analysis: Demonstrate competence to design and conduct investigations and experiments.	5	34 - 42
5. Engineering methods, skills and tools, including information technology: Demonstrate competence to use appropriate engineering methods, skills and tools, including those based on information technology.	4.2	31 - 33
6. Professional and technical communication: Demonstrate competence to communicate effectively, both orally and in writing, with engineering audiences and the community at large.	All	All
9. Independent learning ability: Demonstrate competence to engage in independent learning through well developed learning skills.	1 - 2	1 - 19

Appendix D

Additional tables

Table D.1: Features of the Arduino Mega 2560.

Feature	**Comment**
256k Flash Memory	Neccessary when dealing with large strings of data, especially the responses from AT commands.
16 10-bit Analogue-to-digital ports	Crucial for measuring numerous sensor values.
Four 16-bit Timer/Counter with separate pre-scaler, compare- and capture Mode	Always handy to have a few timers available.
Data retention: 20 years at 85°C/ 100 years at 25°C	This is typically desired for monitoring devices that operate in remote locations.
Master/Slave SPI Serial Interface	Not necessary for a hardware implementation, but it makes it convenient for devices with SPI to send asynchronous interrupts to the Arduino when transmitting data
2-wire Serial Interface or I^2C	

Appendix E

Additional program flow diagrams

This appendix shows all the additional program flow charts that together implements the monitoring software program.

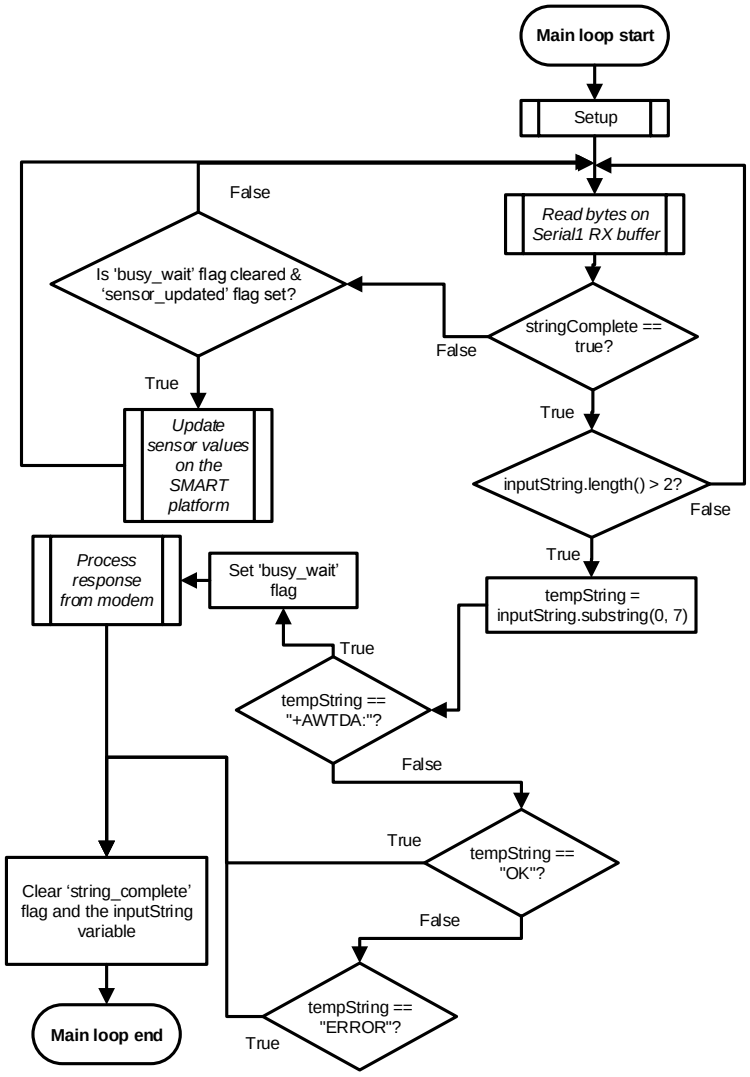


Figure E.1: Flow control diagram of main program.

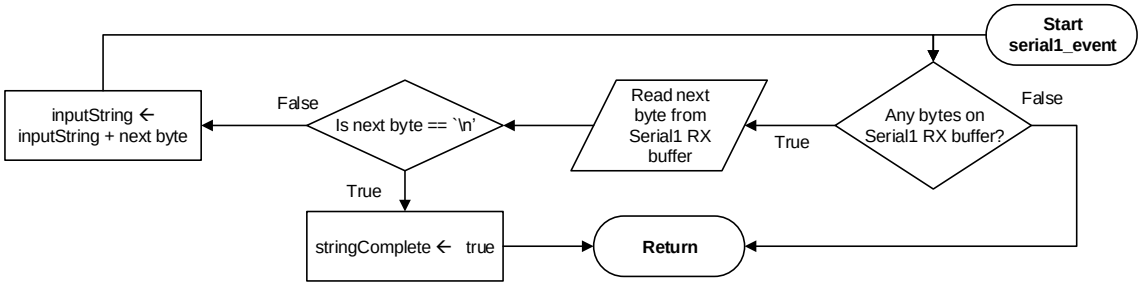


Figure E.2: Flow control diagram for the "serial1_event"-method.

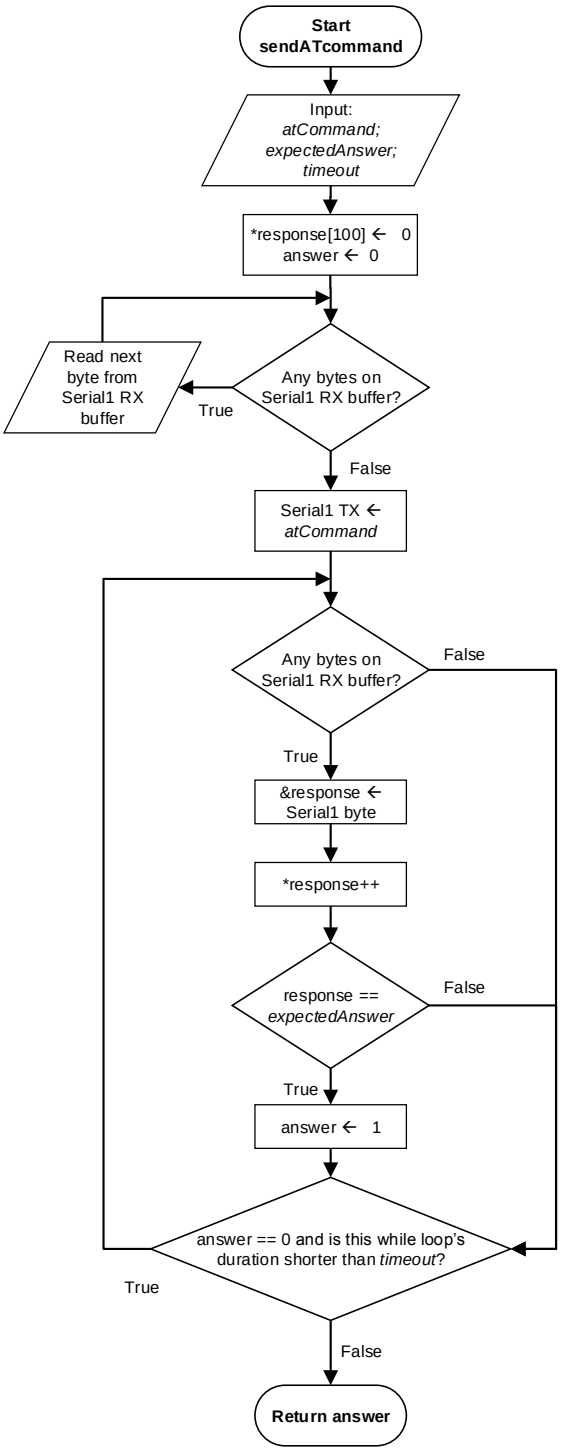


Figure E.3: Flow control diagram of the "sendATCommand"-function.

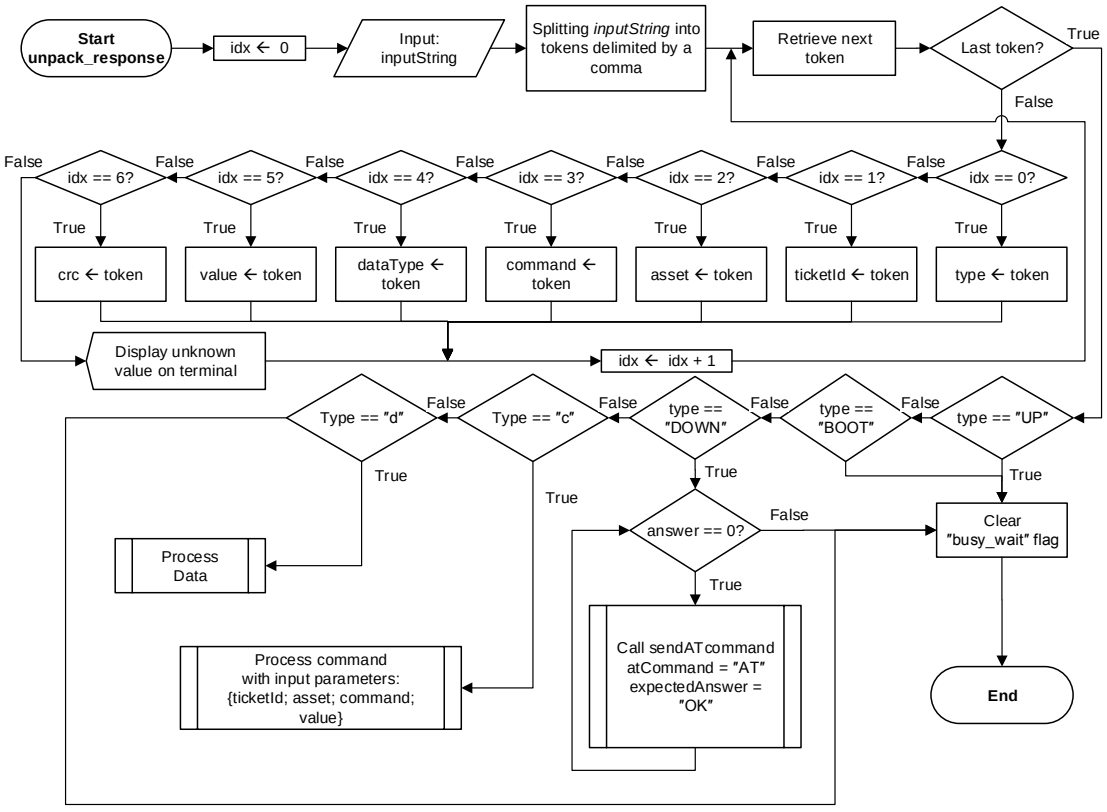


Figure E.4: Flow control diagram of how unsolicited responses from the SMART platform are handled.

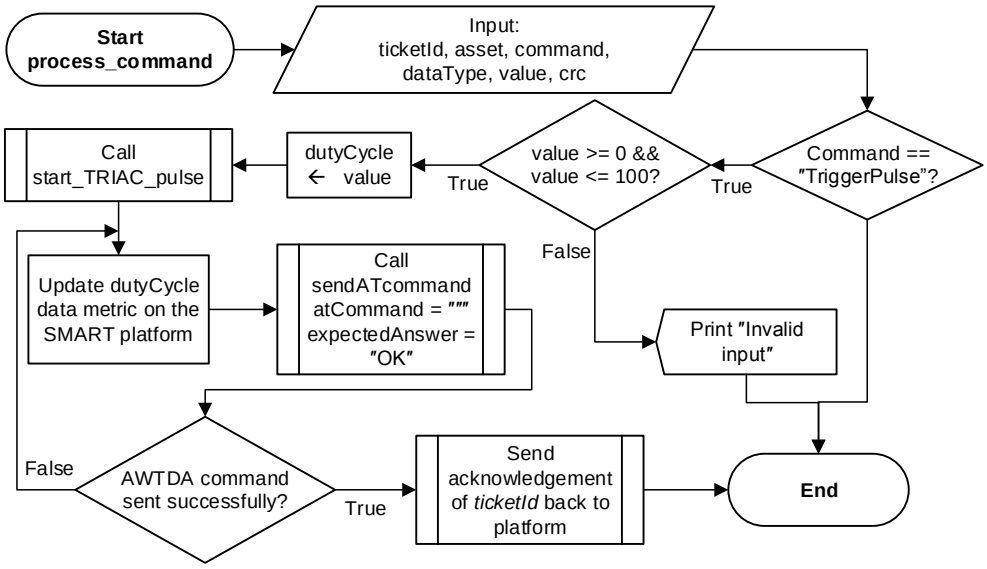


Figure E.5: Flow control diagram for processing the TriggerPulse command.

Appendix F

Bill of materials

In Table F.1 all of the components that were bought for this project is listed. Cost of components such as resistors, capacitors, transistors, wires and veroboards are not listed.

Table F.1: Bill of materials for this project.

Part No.	**Description**	**Manufacturer**	**Unit Price**	**Quantity**	**Cost**	**Supplier**
SCM-202	1.2 L Coffee Percolator	Safeway	R200.00	1	R200.00	Clicks
402	Force-Sensing Resistor (0.5" Circle)	VersaPoint Technology	R84.95	1	R84.95	Netram
UT1812MP	Ultrasound Transducer Transmitter	MEC	R12.98	1	R12.98	Netram
UR1612MP	Ultrasound Transducer Receiver	MEC	R12.98	1	R12.98	Netram
MCP9700	Low-Power Active Thermistor	Microchip	R3.50	2	R7.00	Mantech
MLX90614ESF-BAA	Infrared Thermometer	Melexis	R349.95	1	R349.95	Netram
AX-0500	10A Current Transformer	Nuvotem Talema	R36.23	1	R36.23	RS Components
LM7812C	1A Positive Voltage Regulator	Fairchild	R5.59	2	R11.18	RS Components
VC 10/1/18	PCB Mount 10VA 1x18 o/p	Block	R123.49	1	R123.49	Mantech
MOC3063	Opto-coupled TRIAC DIP	Fairchild	R3.54	1	R3.54	Mantech
BT136-600E	TRIAC TO220 600V 5A	Philips Semiconductors	R3.48	1	R3.48	Mantech
Mega 2560	32-bit AVR-based development board	Arduino	R444.00	1	R444.00	Netram
MAX232N	Dual EIA-232 drivers/receivers	Texas Instruments	R6.40	1	R6.40	Mantech
GBPC104-E4/51	Bridge Rectifier Single Phase 2A 400V	Vishay	R17.73	1	R17.73	RS Components
FM11RF08	RFID 13.56 MHz Book Tag	Fudan Microelectronics	R19.95	2	R39.90	Netram
TL074C	Quad JFET Op-amp	Texas Instruments	R3.88	1	R3.88	Mantech
Totals				**19**	**R1357.69**	

Appendix G

Source code

G.1 Git repositories

All the source code, which includes the test codes used in Chapter , of this project is stored in a git repository hosted on Github

`https://github.com/jeanbritz/RemoteCoffee`

G.2 External Arduino source code

External links listed below provides additional coding examples, which formed part of the programming solution during the system integration phase of this project.

MLX90614 example	`http://bildr.org/2011/02/mlx90614-arduino/`
Atmel AVR126 code	`http://www.atmel.com/Images/AVR126.zip`
AT command example	`http://www.cooking-hacks.com/documentation/tutorials/arduino-3g-gprs-gsm-gps#step15`
AVR strtok_r function	`https://www.inkling.com/read/arduino-cookbook-michael-margolis-2nd/chapter-2/recipe-2-7`
Measuring V_{cc}	`http://provideyourown.com/2012/secret-arduino-voltmeter-measure-battery-voltage/`

Appendix H

Additional diagrams

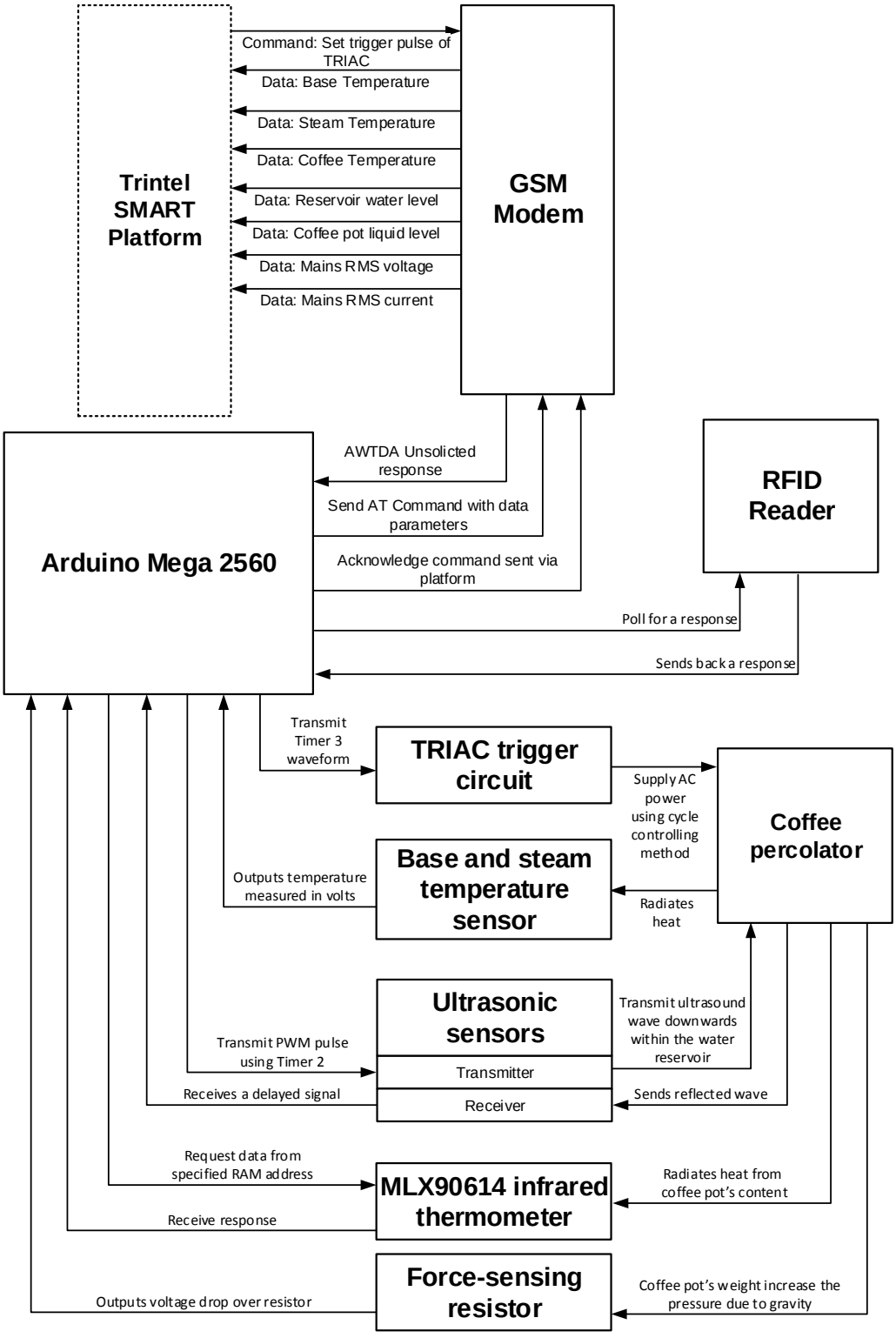


Figure H.1: The flow of data within the system

Appendix I

Additional circuit diagrams

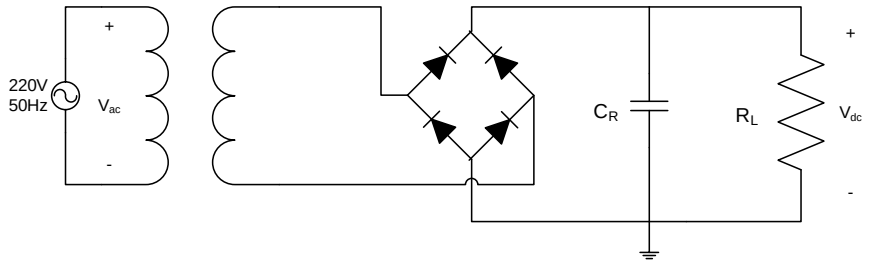


Figure I.1: A full-wave rectifier circuit.

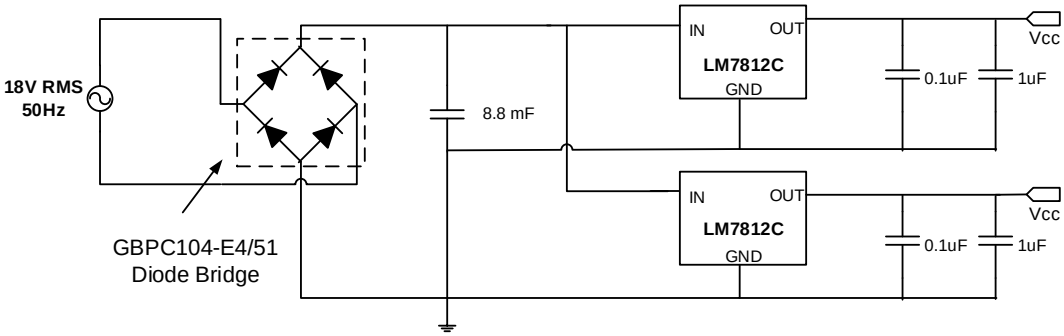


Figure I.2: Circuit designed to be used to build a DC power supply.

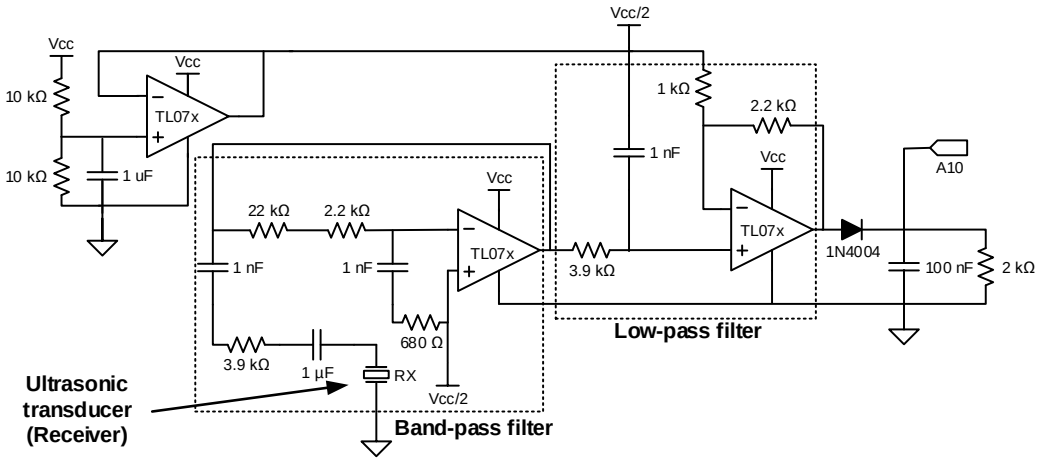


Figure I.3: The circuit for converting the reflected signal to be compatible with the ADC port on the Arduino.

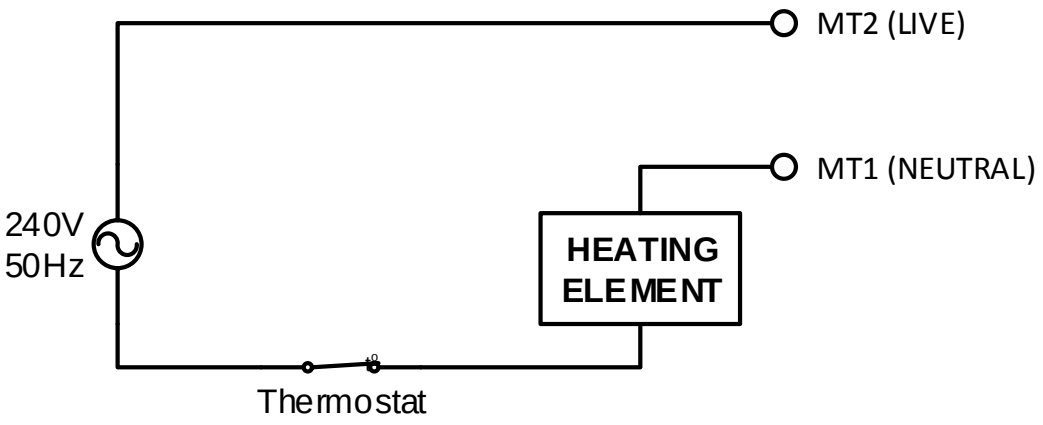


Figure I.4: The circuitry within the percolator.

Appendix J

Interconnections between components

Table J.1: Analogue pin connections between various sensors and the Arduino Mega 2560.

Arduino Pin	**Component Pin**	**Component Description**	**Part no.**
A0	V_{out}	Base temperature sensor	MCP9700
A1	V_{out}	Steam temperature sensor	MCP9700
A2	V_o	Coffee pot's coffee level sensor	-
A3		-	-
A4	-	-	-
A5	-	-	-
A6	-	-	-
A7	-	-	-
A8	V_+	Current sensor	AX-0500
A9	V_o	Voltage sensor	Unknown
A10	V_+	Ultrasonic sensor receiver	-
A11	-	-	-
A12	-	-	-

This appendix supplies additional information of the connections between every component in the system, which is not clear from the circuit diagram. The Arduino is used as the reference component, because about every connection is with one of the Arduino's pins. In Table J.1 the analogue pin configuration is set out. The Arduino Mega has 12 analogue inputs, which is more than required from all the sensors used in this project. The remaining analogue pins are reserved for future use. The part numbers listed here is referring to the list of parts shown in Appendix F. In Table J.1 the part used for measuring voltage is obtained from electronic components that were about to be recycled. It had minimal documentation, but it still operated as expected from an ordinary voltage transformer.

Table J.2: General purpose I/O port connections with various components in the system.

Arduino Pin	**Component Pin**	**Component Description**	**Part no.**
0	TX		
1	RX		
2	Timer 3 Output	Generated waveform is used to trigger the TRIAC switch with a specific duty cycle	MOC3061 and BT136-600E
3	V_+	Timer 2 PWM signal is transmitted to the connected ultrasonic transmitter	UT1812MP
4			
5			
6			
7			
8			
9			
10			
11	Timer 1 Output	Generates a 40 kHz PWM signal that transmits a ultrasound using the ultrasonic transducer	

Appendix K

Test instructions

Table K.1: The first sequence in which the cards are swiped pass the RFID reader.

Step	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Card no.:	1	2	3	4	1	1	2	2	2	3	4	3	3	4

K.1 Test instructions for the temperature sensors test

1. Place the percolator on a level surface with the trigger signal switched off.
2. Pour 920 ml of water into the reservoir.
3. Ensure that the coffee pot is in its proper place (beneath the funnel).
4. Connect the Arduino via its USB-interface to the computer with Arduino IDE version 1.0.2 running.
5. Carefully check the connections between the Arduino and the ones connected to the sensors.
6. Connect the GSM modem's serial port to the Arduino via the RS-232 converter (MAX232). Wait until the GSM modem's status LED to blink at a rate of 200 ms per second.
7. Both the Arduino and the GSM modem must be powered by the 12 V DC power supply.
8. Upload the "test_temp_csv.ino" file onto the Arduino.
9. Open a terminal program (e.g. Serial monitor) and set the baud rate to 9600.
10. Press the reset button on the Arduino to start up the test program.
11. Open a web-browser on the computer and navigate to the `smart.trintel.co.za` website.
12. Type in the login credentials provided and navigate to the "Dashboard".

After the routine is completed the terminal program should show a message stating that the test can be executed by typing the character 'r' and can be stopped by typing the character 's' on the keyboard.

When 'r' is transmitted to the Arduino the program should execute for 10 seconds, before the trigger signal is applied for the percolator to be switched on. Run the test for 5 minutes and refresh the "Dashboard" every 15 seconds to see the metric values being updated. The data logged in the terminal program, displayed in a CSV-format, can be used for data collection.

Table K.2: The second sequence in which the cards are swiped pass the RFID reader.

Step	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Card no.:	4	3	2	1	4	4	4	4	4	3	3	3	1	1

K.2 Test instructions for RFID payment system

1. Place the Adafruit PN532 breakout board on the Arduino.
2. Connect the Arduino via its USB-interface to the computer with Arduino IDE version 1.0.2 running.
3. Connect the GSM modem with a 12 V DC power supply and with the Arduino.
4. Upload the "rfid_test.ino" file onto the Arduino.
5. Open a terminal program (e.g. Serial monitor) and set the baud rate to 9600.
6. Press the reset button on the Arduino to start the test program.
7. Type '1' in the input field of the terminal to format the EEPROM embedded within the Arduino.
8. Type '2' to show the data values currently stored on the EEPROM. If the EEPROM has been formatted correctly then each user's UID would equal to "FF-FF-FF-FF" and all the counters is set to zero.
9. Type '3' to start the main program that implements the payment system.
10. Swipe the cards according to the steps in Table K.1.
11. After the cards have been swiped, press the reset button on the Arduino.
12. Open a web-browser and navigate to the dashboard located at `smart.trintel.co.za` to see the results.
13. Repeat the step 7 and 8. Now use Table K.2 as the order in which the various RFID-cards are being swiped past the RFID reader.
14. Open a web-browser and navigate to the dashboard located at `smart.trintel.co.za` to see the results.

K.3 Test instructions for testing the TRIAC control circuit

1. Plug in the percolator to the mains supply outlet, but do not turn it on yet.

2. Connect the TRIAC trigger circuit to the percolator and the Arduino.
3. Place the percolator on a level surface and connect the oscilloscope's probes to measure pin labelled on the Arduino as D2 and the analogue input pin label A9. Each pin should be measured on different channels.
4. Set the oscilloscope in such a way that both channels' waveforms does not overlap.
5. Connect the Arduino via its USB-interface to the computer with Arduino IDE version 1.0.2 running.
6. Check that the COM port reserved for the Arduino is available on the computer to be used.
7. Carefully check the connections between the Arduino and the ones connected to the voltage sensor.
8. Upload the "actuator_control_test.ino" file onto the Arduino.
9. Open a terminal program (e.g. Serial monitor) and set the baud rate to 9600 (8-N-1).
10. Press the reset button on the Arduino to start up the test program.
11. Start the test by transmitting '0' in the input field located on the terminal's dialogue to set the TRIAC trigger pulse's duty cycle to 0%
12. Log the RMS voltage measurement from the multimeter.
13. Repeat these two steps above by incrementing the input with intervals of 10, 30, 50 and 70%.

K.4 Test instructions for the voltage sensor

1. Connect the voltage sensor's output to A9 on the Arduino.
2. Connect the Arduino via its USB-interface to the computer with Arduino IDE version 1.0.2 running.
3. Connect the voltage sensor to 5V DC supply from the Arduino and connect its ground pin to the Arduino's ground.
4. Connect the mains voltage to the primary side of the voltage transformer. Keep the mains supply switched off.
5. Connect the multimeter in parallel with the mains power supply and set it to measure AC voltages.
6. Upload the "voltage_sensor_test.ino" file onto the Arduino.
7. Open a terminal program (e.g. Serial monitor) and set the baud rate to 9600.

8. Switch the mains supply on.
9. Press the reset button on the Arduino to start up the test program.
10. Log the values displayed on the terminal's window.
11. Log the measurement being showed on the multimeter.
12. Switch the mains supply off.
13. Press the reset button again.
14. Log the values displayed on the terminal's window.
15. Log the measurement being showed on the multimeter.

K.5 Test instructions for testing the ultrasonic sensor

1. Connect the ultrasonic's transmitter to pin 11 on the Arduino and the output of the receive circuit to pin A10/
2. Connect the Arduino via its USB-interface to the computer with Arduino IDE version 1.0.2 running.
3. Upload the "ultra_sensor_test.ino" file onto the Arduino.
4. Open a terminal program (e.g. Serial monitor) and set the baud rate to 9600.
5. Close the lid of the water reservoir.
6. Press the reset button on the Arduino to start up the test program.
7. Log the index value displayed on the terminal's window.
8. Open the lid and pour in 465 ml of water and close the lid.
9. Log the index value displayed on the terminal's window.
10. Open the lid and pour in water so that the total water in the reservoir would be 920 ml of water and close the lid.
11. Log the values displayed on the terminal's window.
12. Open the lid and pour in water until is is full and close the lid.
13. Log the values displayed on the terminal's window.

Appendix L

Additional test data

L.1 Ultrasonic sensor's test data

The following figures is taken before the receiver circuits has been designed. These figures were used to determine how the reflected signal must be scaled to be compatible with the ADC port on the Arduino.

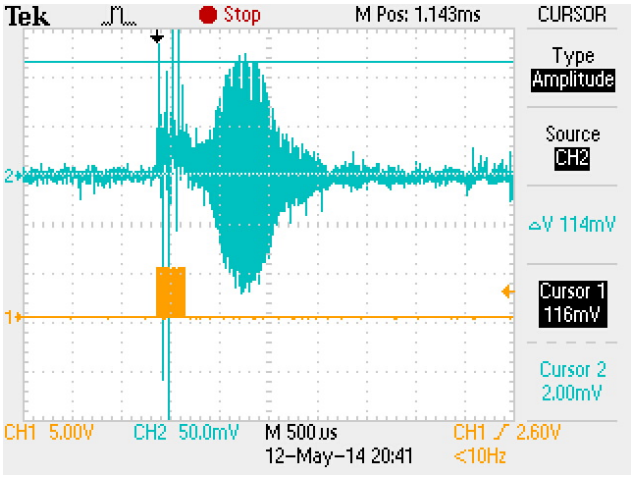


Figure L.1: Measurement of reflected signal's peak voltage when a flat surface is place within 3 centimeters of the ultrasonic transducers.

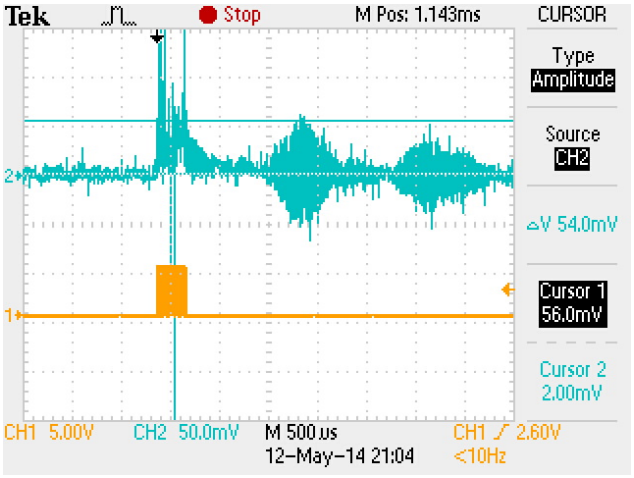


Figure L.2: Measurement of reflected signal's peak voltage when the water resevoir is empty.

L.2 Current sensor's test data

Figure L.3 is used to design the interfacing circuit between the current sensor and the Arduino.

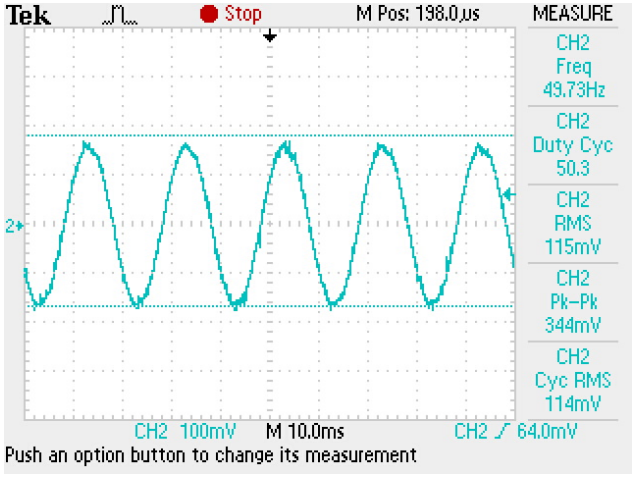


Figure L.3: 50 Hz signal measured by the current sensor.

Appendix M

Test screenshots

M.1 Test results with testing the TRIAC switching circuit

The following figures shows the result obtain during the test of the TRIAC switching circuit.

M.2 Test results of the voltage sensor

The following figure shows the output of the voltage sensor, which were used to determine the RMS value shown in Table 5.3.

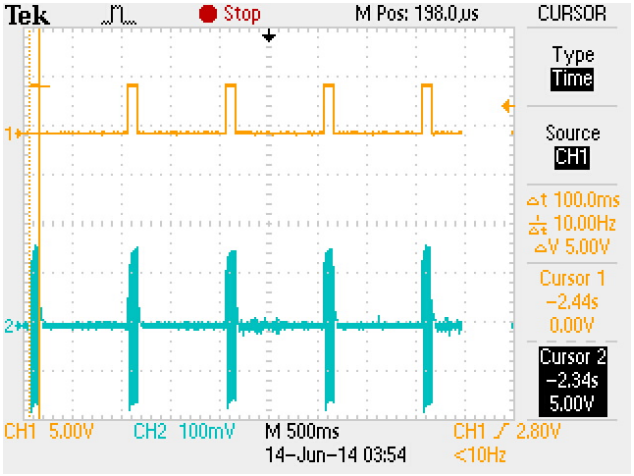


Figure M.1: Screenshot of the oscilloscope with the TRIAC switching with 10% duty cycle.

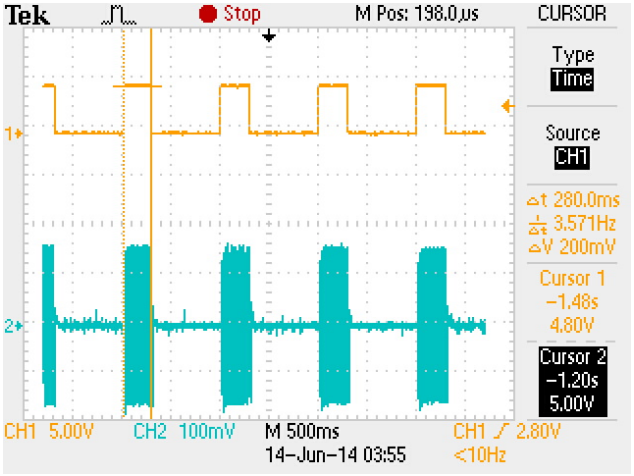


Figure M.2: Screenshot of the oscilloscope with the TRIAC switching with 30% duty cycle.

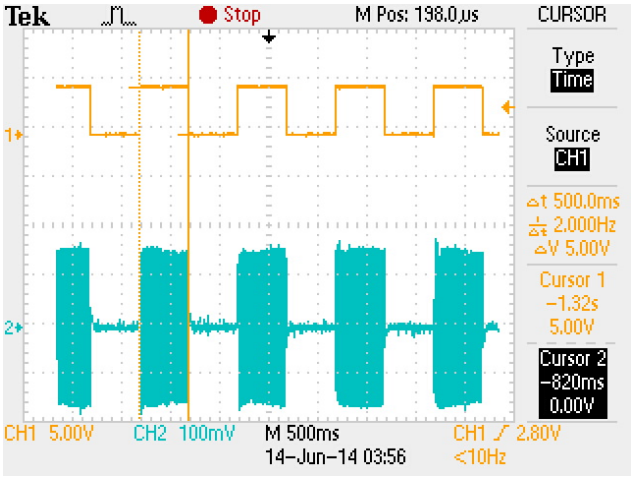


Figure M.3: Screenshot of the oscilloscope with the TRIAC switching with 50% duty cycle.

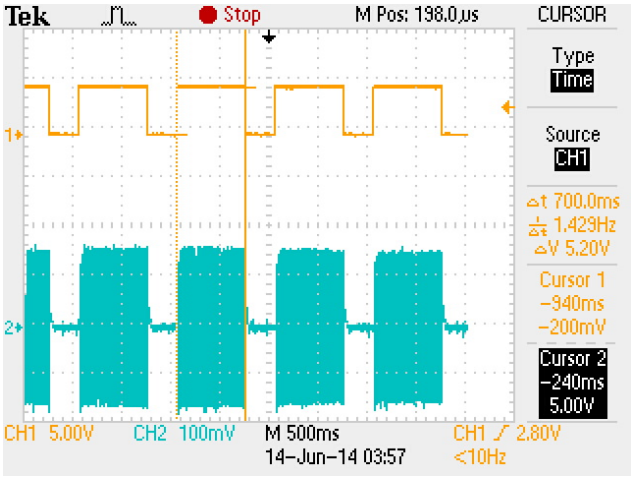


Figure M.4: Screenshot of the oscilloscope with the TRIAC switching with 70% duty cycle.

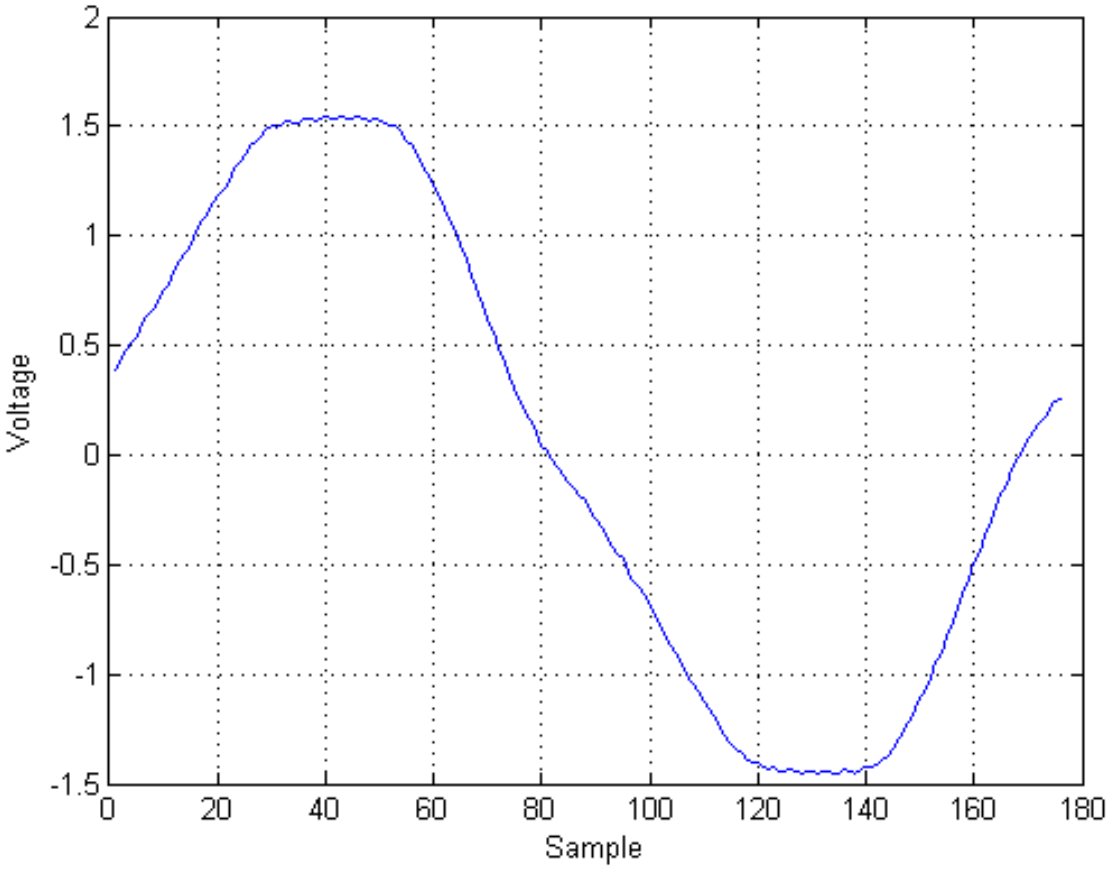


Figure M.5: 50 Hz signal sampled by the Arduino.

M.3 Test results of the RFID payment system

Coffee usage

NAME	AMOUNT
User 1	2 cups
User 2	3 cups
User 3	3 cups

Figure M.6: Screenshot of the coffee usage gadget after the first test for the RFID reader is executed.

Coffee usage

NAME	AMOUNT
User 1	5 cups
User 2	3 cups
User 3	0 cups

Figure M.7: Screenshot of the coffee usage gadget after the second test for the RFID reader is executed.

M.4 Test results of ultrasonic sensor for measuring the reservoir's water level

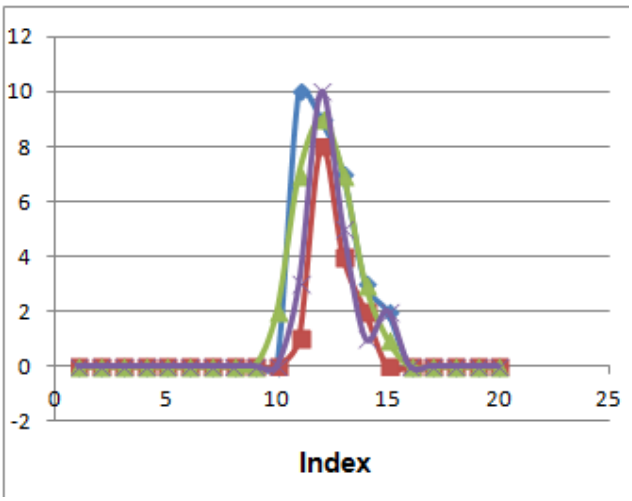


Figure M.8: Plot of ADC values when the water reservoir is empty.

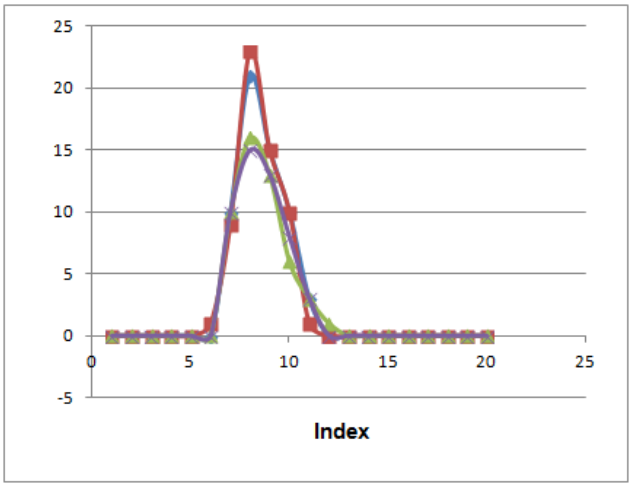


Figure M.9: Plot of ADC value when the water reservoir has 465 ml of water.

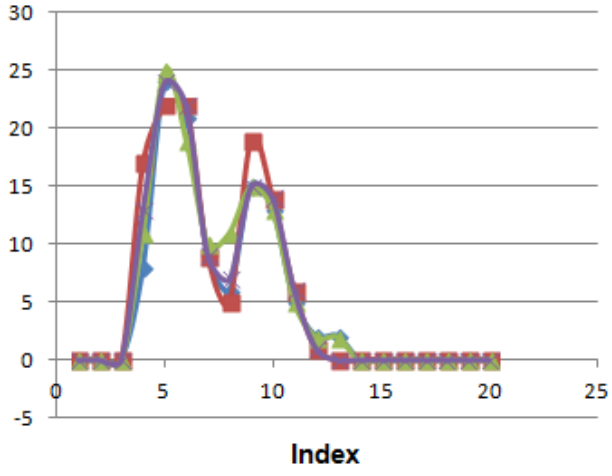


Figure M.10: Plot of ADC value when the water reservoir has 920 ml of water.

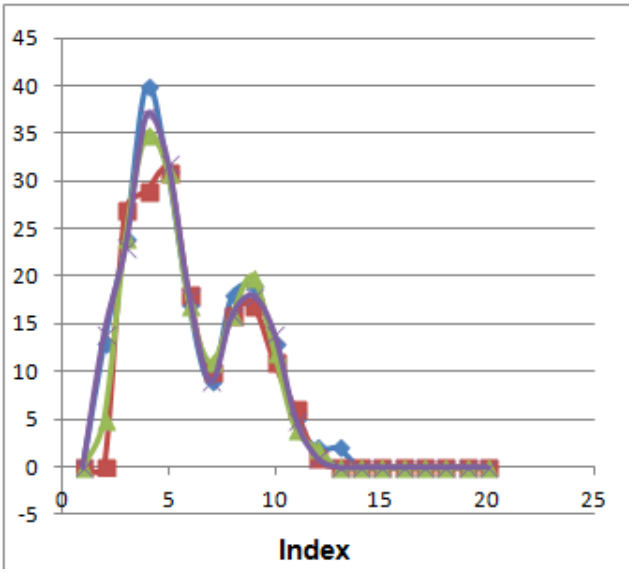


Figure M.11: Plot of ADC value when the water reservoir is full of water.

Appendix N

Test logs

N.1 Terminal output when executing the RFID payment system test

The following sections shows the terminal output when executing the test in section 5.5. First test's log:

```
DEBUG: Coffee usage synchronised with remote platform
Found chip PN532
Firmware ver. 1.6
========================================
        RFID Reader Test
========================================
========================================
| Type '1' to reset the data           |
| Type '2' to view RFID card usage     |
| Type '3' to run the test             |
========================================
========================================
Formatting...Done!
========================================
========================================
| Type '1' to reset the data           |
| Type '2' to view RFID card usage     |
| Type '3' to run the test             |
========================================
========================================
  Unique identity numbers
========================================
User 1 | FF FF FF FF
User 2 | FF FF FF FF
User 3 | FF FF FF FF

Total RFID cards registered : 0
========================================
  Coffee Usage
========================================
User 1 | 0 cups of coffee
User 2 | 0 cups of coffee
User 3 | 0 cups of coffee
========================================
| Type '1' to reset the data           |
| Type '2' to view RFID card usage     |
| Type '3' to run the test             |
========================================
========================================
```

```
 Test the RFID payment system using RFID tags!
==========================================
Found a card -> [ ca:2d:ea:92 ]
INFO: RFID card is added
DEBUG: Coffee usage synchronised with remote platform
Found a card -> [ 2b:b6:eb:63 ]
INFO: RFID card is added
DEBUG: Coffee usage synchronised with remote platform
Found a card -> [ bb:52:f2:63 ]
INFO: RFID card is added
DEBUG: Coffee usage synchronised with remote platform
Found a card -> [ 4d:4d:9e:f7 ]
ERROR: A maximum of 3 RFID cards can be used
Found a card -> [ ca:2d:ea:92 ]
User 1 you can pour yourself a cup of coffee.
Total coffee used : 1 cups
DEBUG: Coffee usage synchronised with remote platform
Found a card -> [ ca:2d:ea:92 ]
User 1 you can pour yourself a cup of coffee.
Total coffee used : 2 cups
DEBUG: Coffee usage synchronised with remote platform
Found a card -> [ 2b:b6:eb:63 ]
User 2 you can pour yourself a cup of coffee.
Total coffee used : 1 cups
DEBUG: Coffee usage synchronised with remote platform
Found a card -> [ 2b:b6:eb:63 ]
User 2 you can pour yourself a cup of coffee.
Total coffee used : 2 cups
DEBUG: Coffee usage synchronised with remote platform
Found a card -> [ 2b:b6:eb:63 ]
User 2 you can pour yourself a cup of coffee.
Total coffee used : 3 cups
DEBUG: Coffee usage synchronised with remote platform
Found a card -> [ bb:52:f2:63 ]
User 3 you can pour yourself a cup of coffee.
Total coffee used : 1 cups
DEBUG: Coffee usage synchronised with remote platform
Found a card -> [ 4d:4d:9e:f7 ]
ERROR: A maximum of 3 RFID cards can be used
Found a card -> [ bb:52:f2:63 ]
User 3 you can pour yourself a cup of coffee.
Total coffee used : 2 cups
DEBUG: Coffee usage synchronised with remote platform
Found a card -> [ bb:52:f2:63 ]
User 3 you can pour yourself a cup of coffee.
```

```
Total coffee used : 3 cups
DEBUG: Coffee usage synchronised with remote platform
Found a card -> [ 4d:4d:9e:f7 ]
ERROR: A maximum of 3 RFID cards can be used
```

Second test's log:

```
DEBUG: Coffee usage synchronised with remote platform
Found chip PN532
Firmware ver. 1.6
========================================
        RFID Reader Test
========================================
========================================
| Type '1' to reset the data           |
| Type '2' to view RFID card usage     |
| Type '3' to run the test             |
========================================
========================================
Formatting...Done!
========================================
========================================
| Type '1' to reset the data           |
| Type '2' to view RFID card usage     |
| Type '3' to run the test             |
========================================
Found chip PN532
Firmware ver. 1.6
========================================
        RFID Reader Test
========================================
========================================
| Type '1' to reset the data           |
| Type '2' to view RFID card usage     |
| Type '3' to run the test             |
========================================
========================================
 Test the RFID payment system using RFID tags!
========================================
Found a card -> [ 4d:4d:9e:f7 ]
INFO: RFID card is added
DEBUG: Coffee usage synchronised with remote platform
Found a card -> [ bb:52:f2:63 ]
INFO: RFID card is added
DEBUG: Coffee usage synchronised with remote platform
Found a card -> [ 2b:b6:eb:63 ]
```

```
INFO: RFID card is added
DEBUG: Coffee usage synchronised with remote platform
Found a card -> [ ca:2d:ea:92 ]
ERROR: A maximum of 3 RFID cards can be used
Found a card -> [ 4d:4d:9e:f7 ]
User 1 you can pour yourself a cup of coffee.
Total coffee used : 1 cups
DEBUG: Coffee usage synchronised with remote platform
Found a card -> [ 4d:4d:9e:f7 ]
User 1 you can pour yourself a cup of coffee.
Total coffee used : 2 cups
DEBUG: Coffee usage synchronised with remote platform
Found a card -> [ 4d:4d:9e:f7 ]
User 1 you can pour yourself a cup of coffee.
Total coffee used : 3 cups
DEBUG: Coffee usage synchronised with remote platform
Found a card -> [ 4d:4d:9e:f7 ]
User 1 you can pour yourself a cup of coffee.
Total coffee used : 4 cups
DEBUG: Coffee usage synchronised with remote platform
Found a card -> [ 4d:4d:9e:f7 ]
User 1 you can pour yourself a cup of coffee.
Total coffee used : 5 cups
DEBUG: Coffee usage synchronised with remote platform
Found a card -> [ bb:52:f2:63 ]
User 2 you can pour yourself a cup of coffee.
Total coffee used : 1 cups
DEBUG: Coffee usage synchronised with remote platform
Found a card -> [ bb:52:f2:63 ]
User 2 you can pour yourself a cup of coffee.
Total coffee used : 2 cups
DEBUG: Coffee usage synchronised with remote platform
Found a card -> [ bb:52:f2:63 ]
User 2 you can pour yourself a cup of coffee.
Total coffee used : 3 cups
DEBUG: Coffee usage synchronised with remote platform
Found a card -> [ ca:2d:ea:92 ]
ERROR: A maximum of 3 RFID cards can be used
Found a card -> [ ca:2d:ea:92 ]
ERROR: A maximum of 3 RFID cards can be used
```