ECC3304 [Group 1] Digital Systems Design

SEMESTER I 2021/2022

PROJECT REPORT

Smart cities: Digital solutions for a more livable future

GROUP 5 NEOTECH: PACQED	
NAME	**MATRIC NO.**
TABINA KAMAL	208651
MUHAMMAD AFIQ IQHMAL BIN SYAMSUL KAMAL	208182
NIK AKIF HAZIQI BIN ROHAIDI	208150
SITI NOR HANIM BINTI JEMIAN	206219

Submission Date: 02 February 2022

Lecturer: PUAN ROSLIZAH BINTI ALI

CONTENTS

1. INTRODUCTION

A. Background of Project

A smart city employs information and communication technology (ICT) to boost operational efficiency, share data with the public, and improve government service and citizen welfare [1]. The goal is to improve the quality of life of the people living in it. In this design project, the members ‘role-played’ as a start-up company ‘NeoTech’ and designed a product called 'PACQED'. This product is a handbag that is able to combat the unfavorable climatic conditions in Malaysia for the storage of cosmetic products and medication.

This product will monitor and regulate the temperature and humidity levels inside the handbag, ensuring optimum storage conditions for contents in the bag, thus retaining their quality. This will reduce the chances of negative effects such as allergic reactions and ineffectivity of the medication or cosmetic product [2][3]. Therefore, ‘PACQED’ would be suited to the concept of a ‘smart city’ as it solves the problem of the degradation of cosmetics and medication in handbags.

A market survey was then carried out to find out the relevance of such a product as ‘PACQED’. The survey was divided into two categories; primary and secondary research. The primary research consisted of online surveys and on-site interviews for both quantitative and qualitative information from the target group, who are adult women. The secondary research consisted of literature review and online research. The sources used were; articles, journals, websites and reports. These provided statistics and facts relating to the issue.

B. Project Scope and Specifications

For the purpose of this project, selected functionalities of the product; ‘PACQED’, were implemented onto the DE1-SoC board. The objectives of ‘PACQED’ are the following; monitoring and regulating environmental conditions in the handbag and ensuring that contents (cosmetic products and medication) in the handbag are kept at optimal storage conditions.

The functionalities selected were; monitoring and regulating temperature and humidity conditions. For the purpose of monitoring the mentioned conditions, a ‘2-in-1’ temperature and humidity sensor (DHT22) was used. For the regulation of these conditions, the following were used; dual-way fan using DC motor and a peltier cooler. A motor driver (L298N) was used to control the direction of rotation of the fan as well as an intermediary between the components and DE1 board. The DHT22 sensor was the input and was connected to an Arduino board in order to receive data. The Arduino board was to be interfaced with the DE1 board in order to run the other components automatically according to the detected changes. The peltier cooler was operated with regards to the changes in temperature and the dual-way fan operated with changes in humidity levels. Therefore, the product objectives would be able to be shown on the DE1 board.

Verilog entry was used to achieve the above goals. Hierarchical design was utilised in order to separate the different functionalities into modules. The behavioural design method was used in this project. All the modules were compiled into one top-level module which is called ‘pacqed’. The design was then programmed onto the DE1-SoC board.

C. Feature Set

Module	Hardware	Software
CoolerTest	Peltier cooler	Intel Quartus Prime 20.1 lite edition
MotorTest	Dual-way fan [DC Motor]	
-	Motor driver (L298N)	
-	Temperature and humidity sensor (DHT22)	

Table A: List of modules according to its hardware and software.

2. MANAGEMENT

A. Roles of Members

	NAME/POSITION	JOB FUNCTION
1	 **TABINA KAMAL**	• Leader of the group • Wrote the verilog code • Designed the circuit • Role play position: Chief Executive Officer • Proposal- slide editor (CEO) • Report- editor • Pitching - presenter
2	 **NIK AKIF HAZIQI**	• Designed the circuit • Wrote the verilog code • Role play position : Chief Technical Officer • Proposal - slide editor (CTO) • Report - editor • Pitching - slide editor
3	 **AFIQ IQHMAL**	• Wrote the verilog code • Role play position: Chief Marketing Officer • Proposal - slide editor (CMO) • Report - editor • Pitching - slide editor

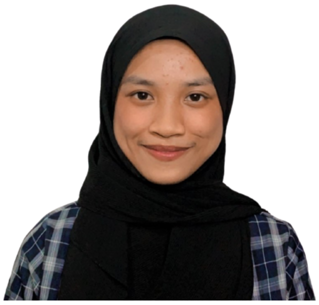
4	 **SITI NOR HANIM**	• Wrote the verilog code • Role play position: Chief Financial Officer • Proposal - slide editor (CFO) • Report - editor • Pitching- slide editor

Table B: Roles of group members

B. Project Risks

1. Technical and Operational Risk

Technology-related risks are rather complicated to manage since implementing new IT systems requires additional training and software procurement and it might lead to delays and project failure. Also, the project may be delayed or terminated if vital operations and fundamental procedures are not implemented properly.

2. Cost Risk

A shortage or mismanagement of project funding as a consequence of an inflated budget or other limitations endangers the completion of the project.

3. Health and Safety Risk

Health and safety standards must be monitored and evaluated on a regular basis in order to identify any risks that might result in losses or fines for the user or company.

4. Market Risk

Market risk is likely to occur when a project fails to deliver the expected outcomes. Competitors may seize the opportunity to cripple and eliminate the business from the market. Liquidity, credit, and interest rate fluctuations are additional possible market distractions for the project's product sales.

C. Project Cost

No.	Item	Quantity	Price / RM
1	L298N DC Motor Driver	1	10.00
2	DHT22 Temperature and Humidity Sensor	1	22.00
3	Arduino UNO R3 Board	1	45.00
4	Jumper Cables (x5 Sets)	1	10.00
5	Crocodile Cables	1	2.00
6	Peltier Cooler Set - Cooler [RM 18] - Heat Sink + Paste [RM 13]	1	31.00
		TOTAL AMOUNT	120.00

Table C: Actual cost of components purchased

No.	Item	Quantity
1	Terasic DE1-SoC Board	1
2	DC Motor	1

Table D: Components supplied by department

D. Project Progression

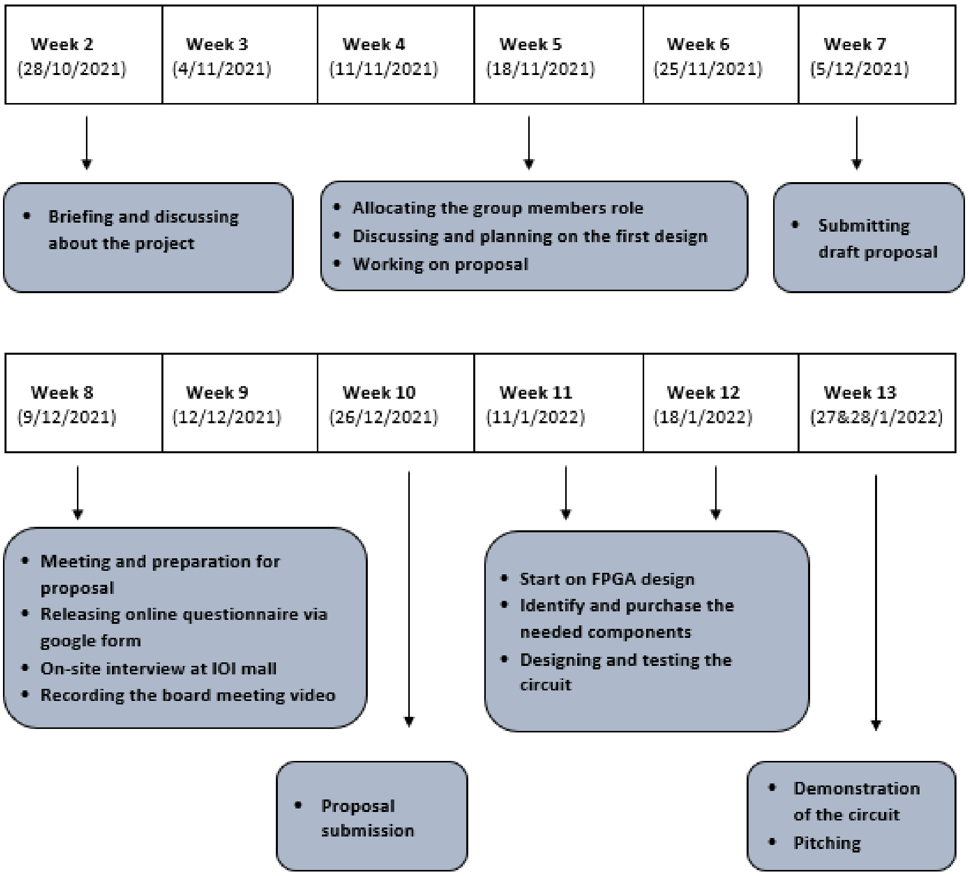


Figure A: Project milestones within 14 weeks

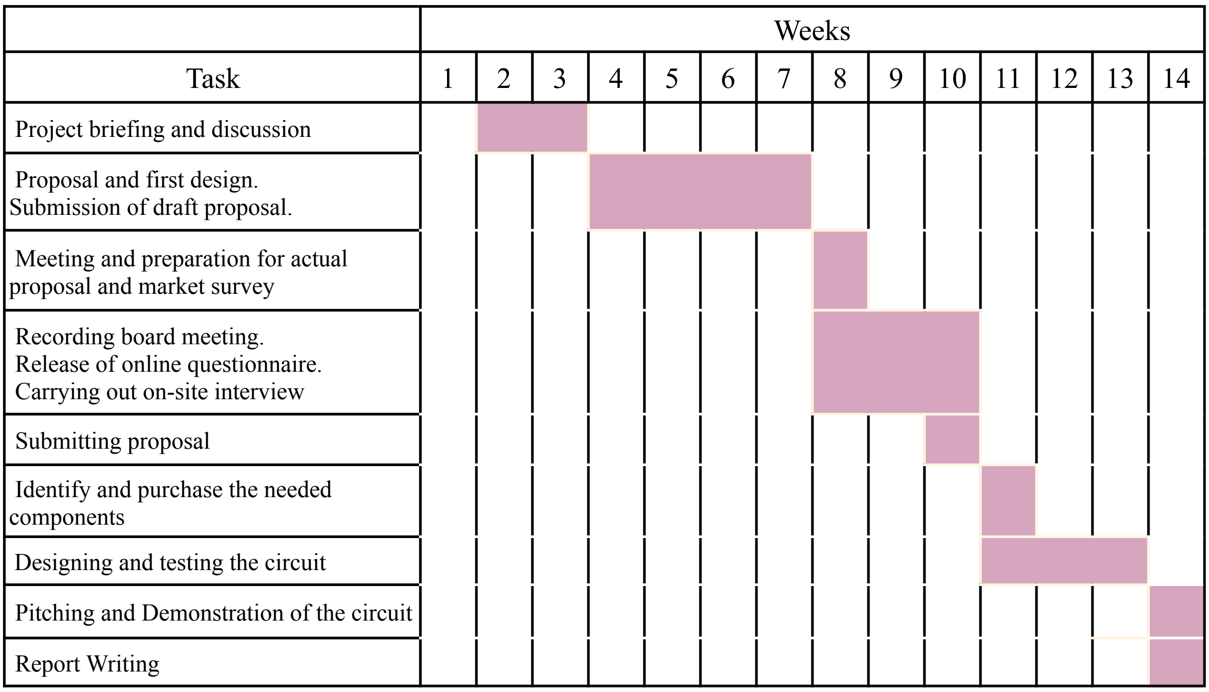


Figure B: Gantt chart of project activities

3. METHODOLOGY

A. Design Approach

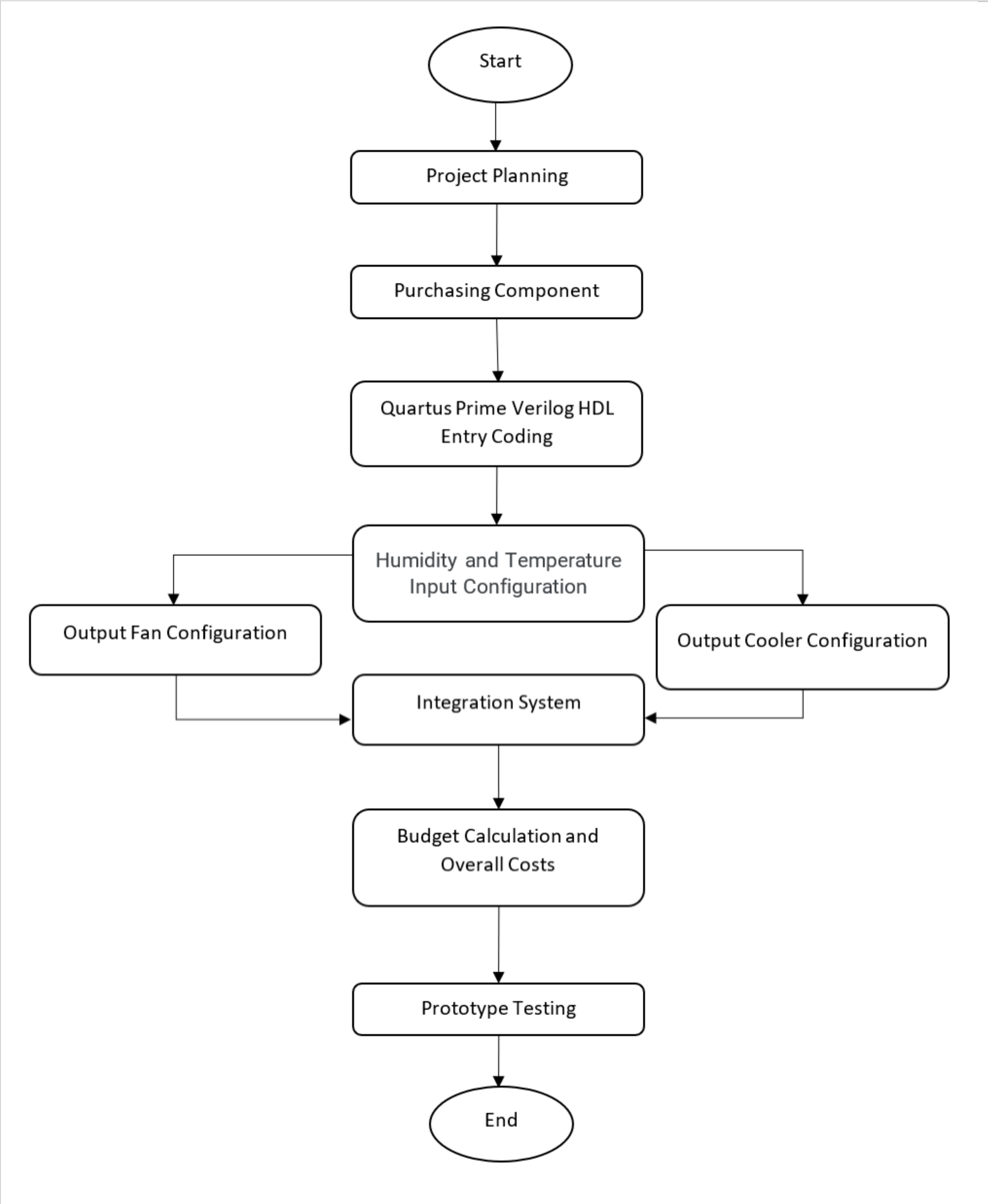


Figure 1: Overall Design Approach

B. Design Flow

The design flow of the ‘PAQCED’ module begins with obtaining inputs of temperature and humidity levels from the DHT22 temperature and humidity sensor that is connected to an Arduino UNO board. The sensor will continuously monitor and send inputs to the Arduino board which is connected to the DE1-SoC board whereby the DE1 board receives the temperature and humidity data from the Arduino board. The motor driver (L298N) was used to control the direction of rotation of the fan. It also was an intermediary between the components and DE1 board; the motor (3 - 9V) and peltier cooler (6 - 12V) required a higher voltage than what was supplied form the board (5V).

The different inputs from the sensors were assigned to states so each state would carry out a specific function. The peltier cooler and the dual-way fan function simultaneously. When the temperature is at an acceptable range (00), the peltier cooler will stay off. However, when the temperature goes above the acceptable threshold (11), then the peltier cooler will be turned on. When the humidity level is high (100), the dual way fan will bring the air from inside to outside (FAN OUT). However, when the humidity level is low, the dual way fan will bring the outside air into the inside of the bag (FAN IN). The ‘FAN IN’ and ‘FAN OUT’ states cause the fan to rotate in opposite directions. The dual way fan will be off when the humidity level is at an acceptable range (010). Therefore, this cycle of obtaining inputs and producing outputs is repeated to keep the quality of the contents inside of the bag at optimum conditions.

All the inputs and outputs would be implemented onto the DE1-SoC FPGA board by using different modules such as “Top Level” module which acts as a wrapper to the “Dual-way Fan" module and “Cooler” module. Port module instantiation was used for the purpose of ‘linking’ the modules together. The Verilog entry and behavioural design method was used in this project. The functional waveform simulation was carried out using Quartus Prime software. The timing simulation was not possible as Cyclone V does not support the timing simulation. Ultimately, the design was then programmed onto the DE1-SoC board and tested.

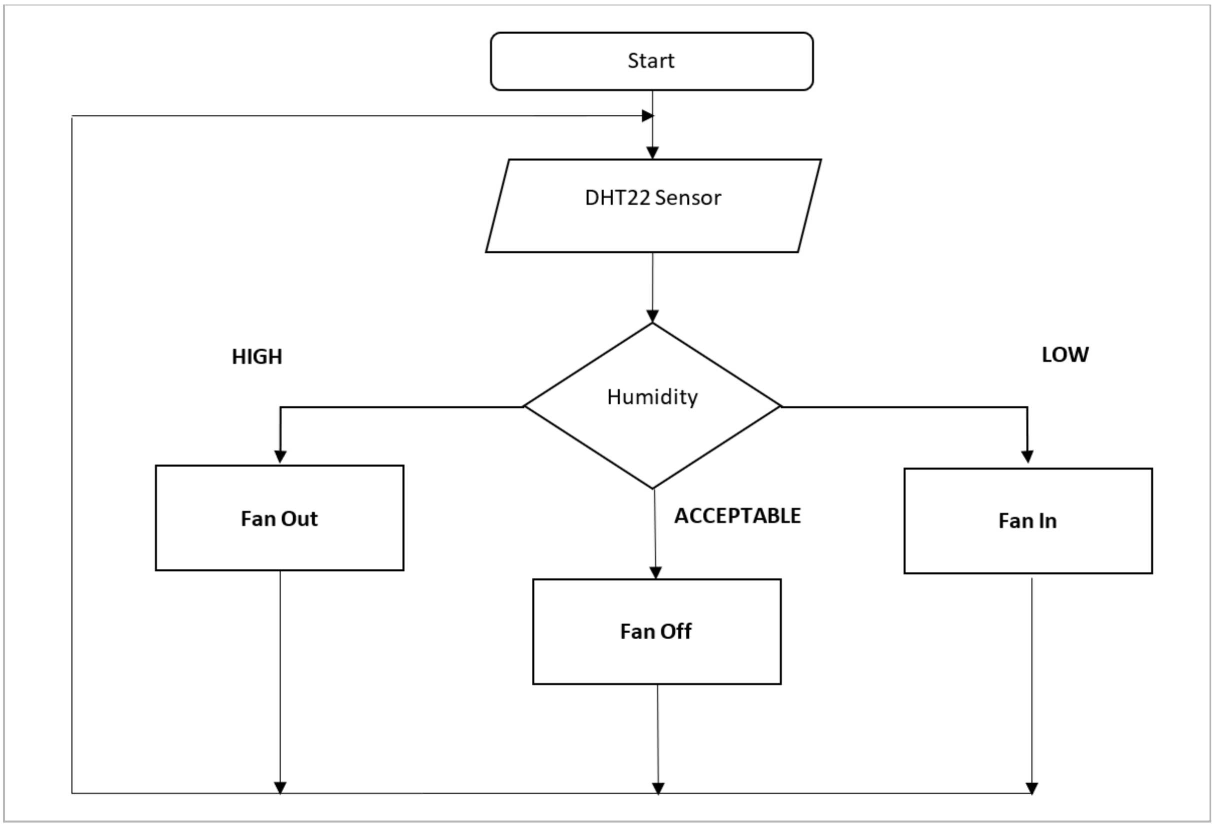


Figure 2: Flow chart of Dual-way Fan module (Motortest)

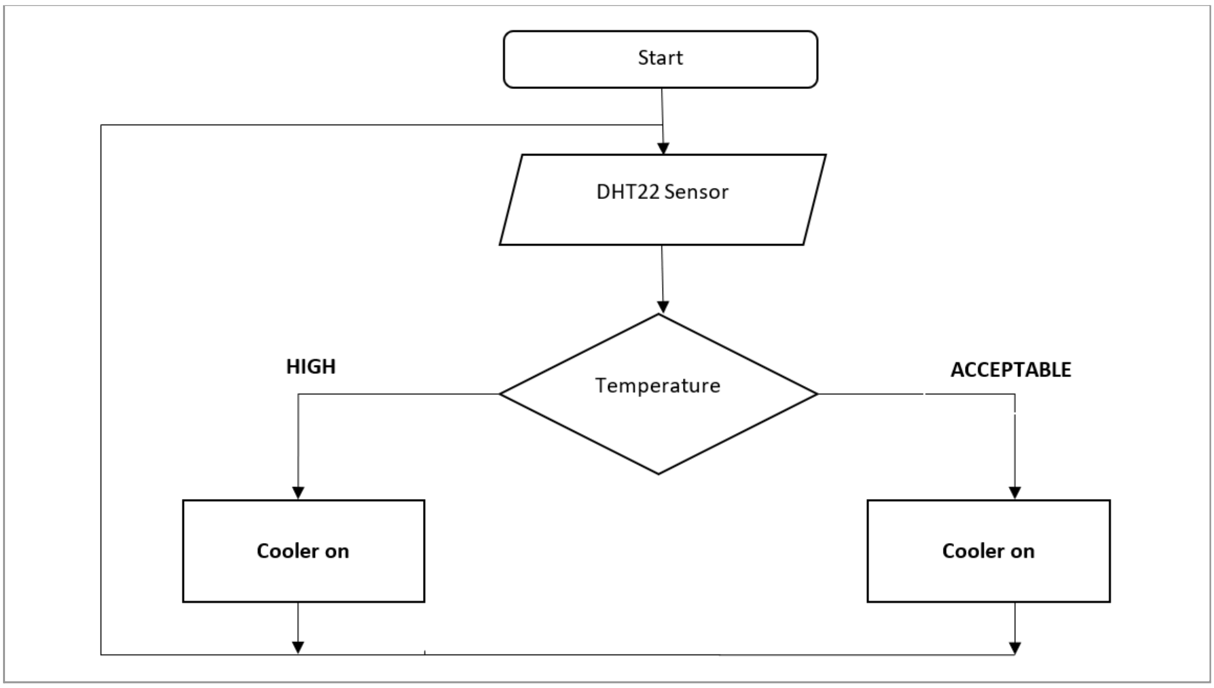


Figure 3: Flow chart of Cooler module (CoolerTest)

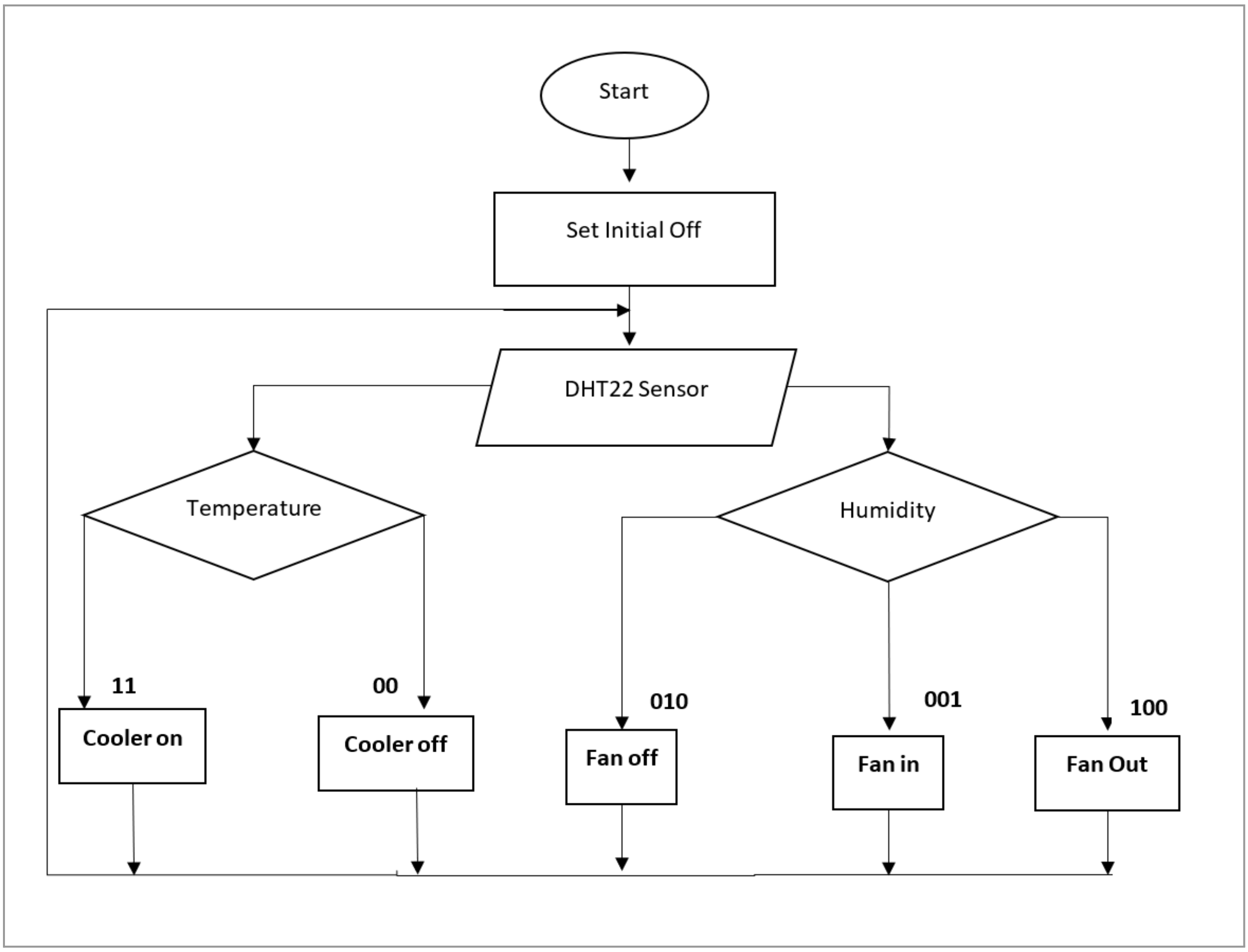


Figure 4: Overall flowchart for pacqed

4. DESCRIPTION

Block diagram

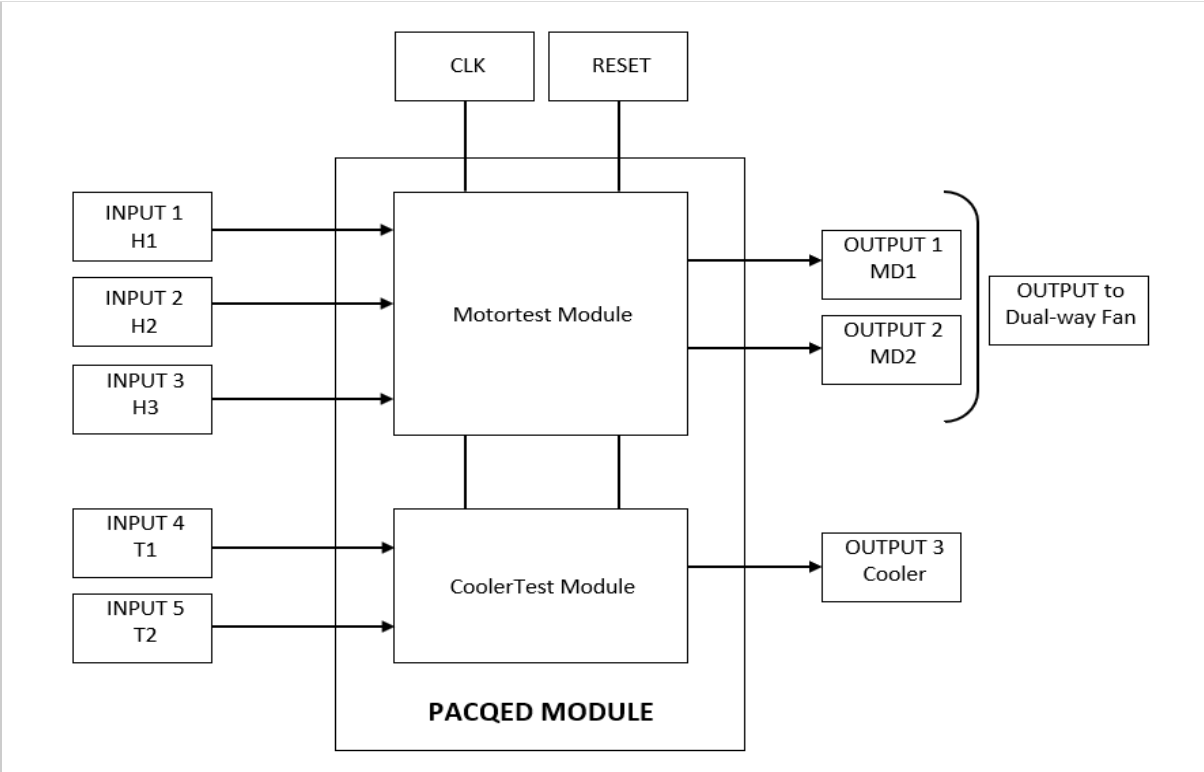


Figure 5 : Block diagram of 'pacqed' (Initial Design)

The initial design was supposed to use the values of {H1, H2, H3} and {T1, T2} which are the humidity and temperature inputs from the Arduino board obtained from the DHT22 sensor, respectively. The outputs MD1 and MD2 are based on inputs H1, H2 and H3. The outputs of MD1 and MD2 determine the rotation direction of the dual-way fan. While the output for Cooler is based on inputs T1 and T2.

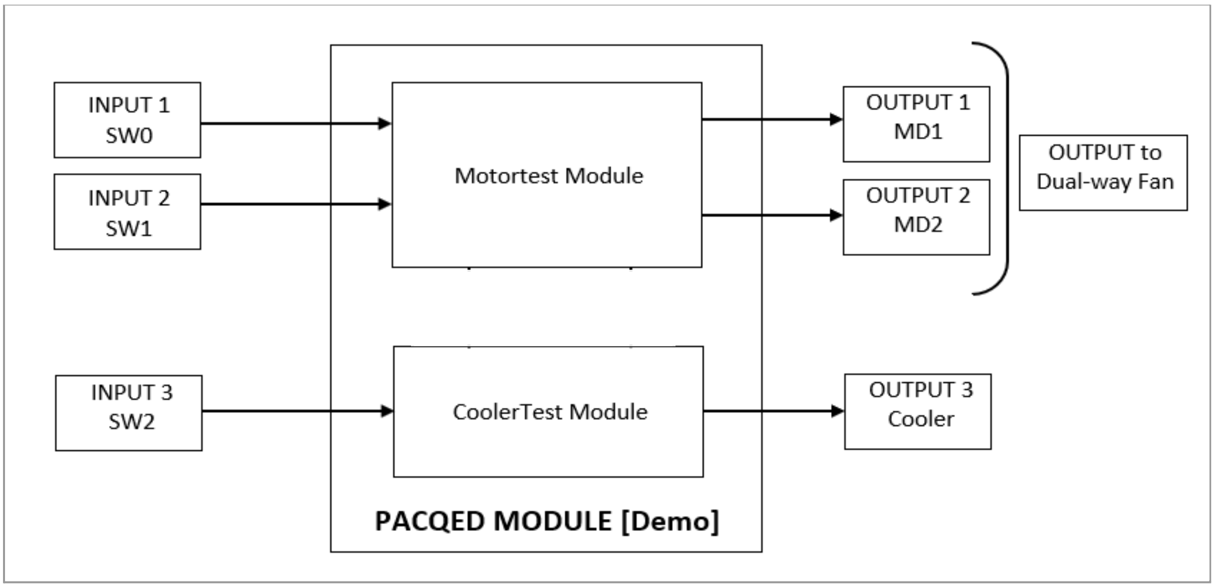


Figure 6 : Block diagram of Pacqed (Demonstrated)

From Figure 6 above, the actual design that was implemented on the DE1-SoC FPGA board, utilised switches to represent the inputs instead of the inputs from the Arduino from the DHT22 sensor. The outputs are still the same as that of the initial design; MD1, MD2 and Cooler.

State diagrams

The state diagrams obtained are from the initial design, which is a sequential circuit. The demonstrated design however, was a combinational circuit therefore, it does not have state diagrams.

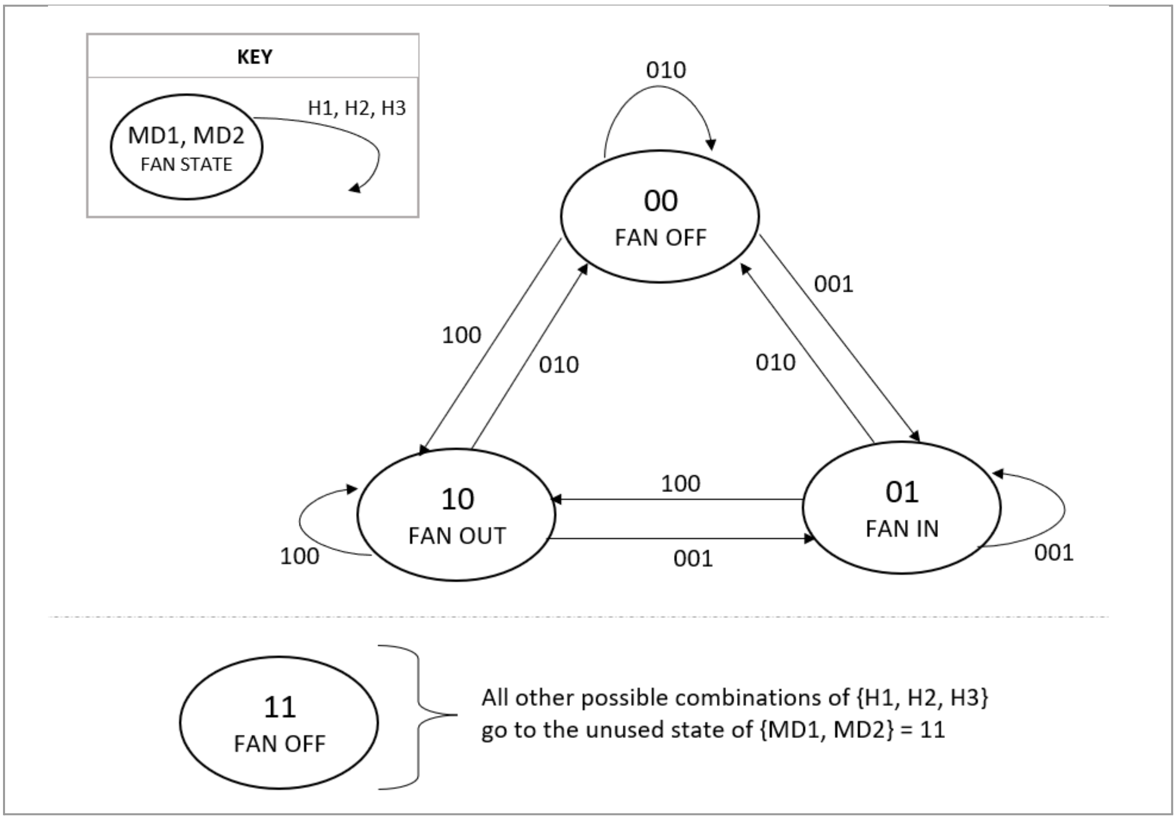


Figure 7 : State diagram for dual-way fan operation [Motortest Module - Initial Design]

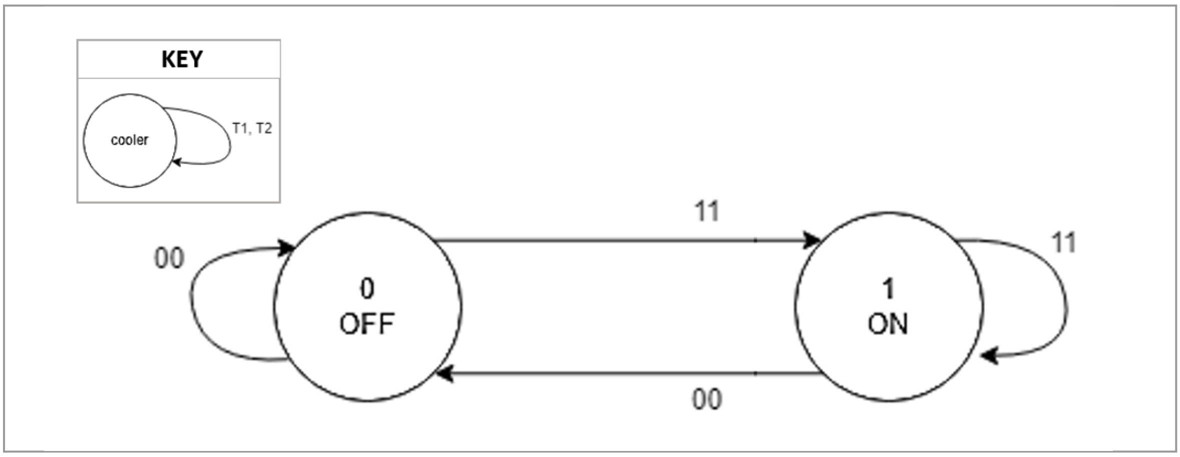


Figure 8 : State diagram for cooler operation [CoolerTest Module - Initial Design]

Description of top-level design

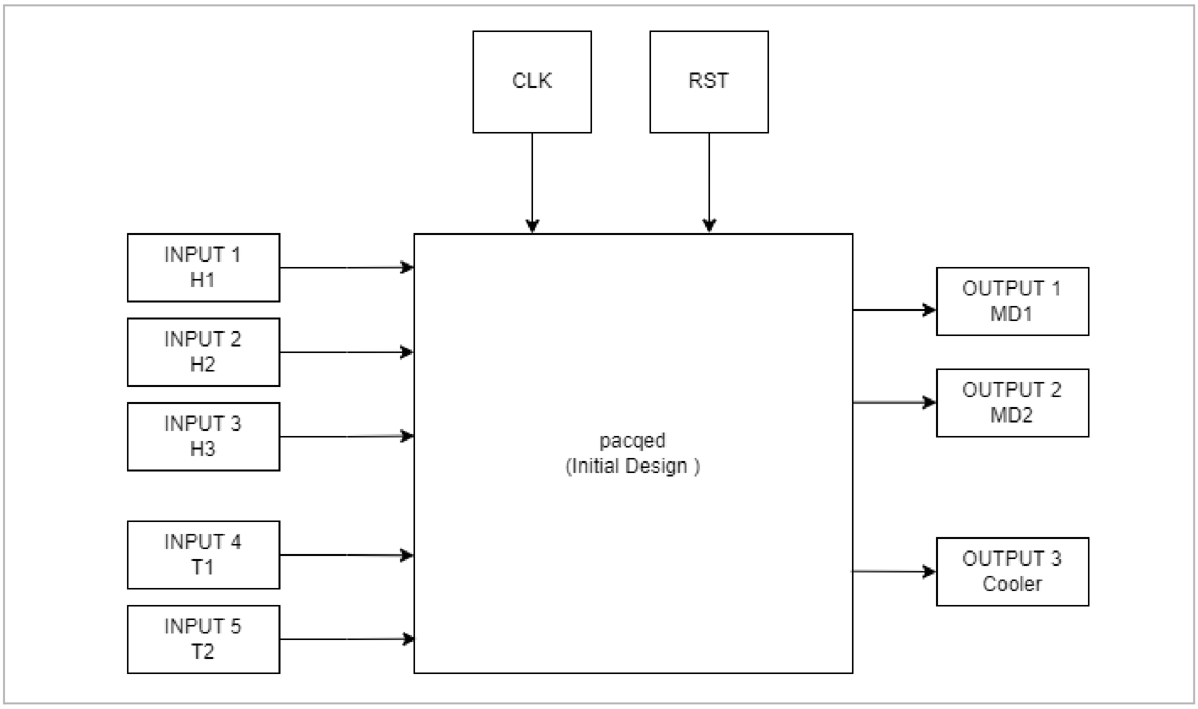


Figure 9: 'pacqed' top-level block diagram [Initial design]

From Figure 9, 'pacqed' is the top-level module of the initial design that takes in five inputs. Three of them (H1, H2, H3) correspond to humidity levels and the remaining two inputs (T1, T2) correspond to temperature. There are three outputs. Outputs MD1 and MD2 operate the dual-way fan and the Cooler output operates the peltier cooler. The initial design also takes in a clock and has an asynchronous reset. Figure 10 below shows the top-level diagram of the demonstrated circuit which utilises switches as inputs. SW0 and SW1 correspond to humidity levels and SW2 corresponds to temperature levels. The outputs are the same as that of the initial design.

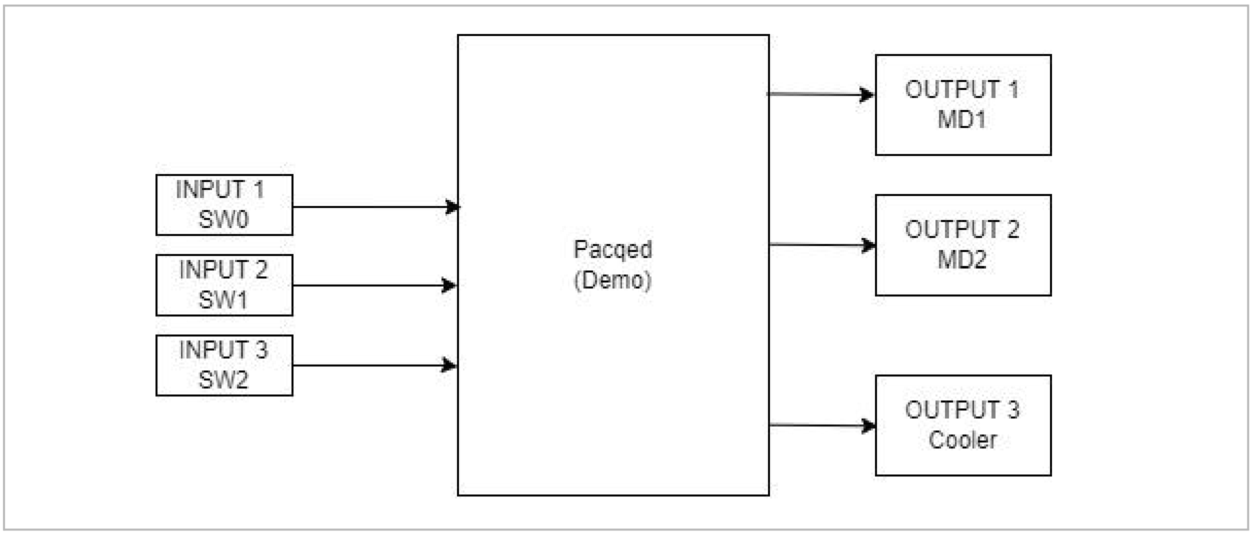


Figure 10: 'pacqed' top-level block diagram [Demonstrated]

Description of sub-modules

A) Motortest module

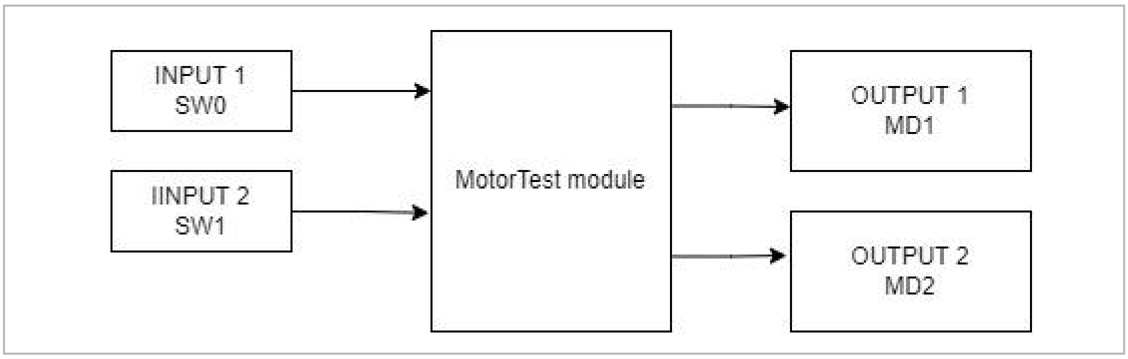


Figure 11 : MotorTest module [Demonstrated]

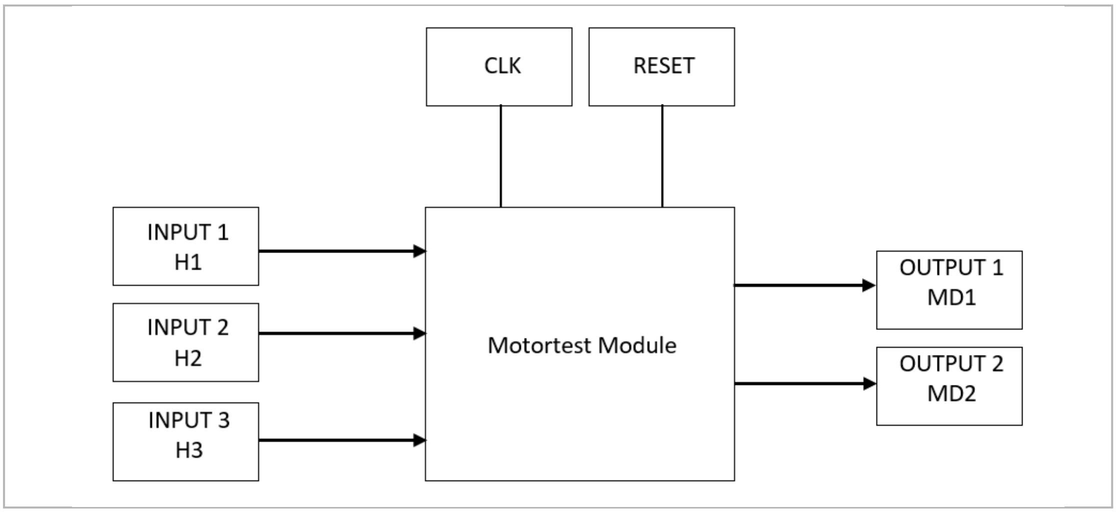


Figure 12 : MotorTest module [Initial Design]

Figure 11 above, shows the block diagram of the demonstrated Motortest module. The inputs were switches; SW0 and SW1. The outputs were MD1 and MD2, which were connected via the GPIO output pins to the L289N Motor Driver which operated the dual-way fan. The outputs MD1 and MD2 would be turned HIGH and LOW based on the combinational input of the switches. When {SW0, SW1}= 00 or {SW0, SW1}= 11, the fan would be off. When {SW0, SW1}= 01, the fan would turn on to be in the 'FAN IN' state and when {SW0, SW1}= 10, the fan would rotate in the opposite direction in the 'FAN OUT' state. According to the truth table found in Part 5, the dual-way fan shows the same functionality as the XOR function.

Figure 12 shows the block diagram of the initial design of the Motortest module. The outputs to be obtained are exactly the same as in Figure 11, however the inputs are different. The inputs H1, H2 and H3 were intended to arrive from the Arduino UNO board. For testing purposes three humidity ranges were identified and assigned to three different states of {H1. H2. H3}. With reference to the Arduino code in Section

3 of the Appendix; {H1, H2, H3} = 001 represented a low humidity level of 30% and below which would incur the 'FAN IN' output, {H1, H2, H3} = 010 represented an acceptable humidity range from 30-80% which would incur the 'FAN OFF' output and lastly {H1, H2, H3} = 100 represented a high humidity level of 80% and above which would incur the 'FAN OUT' output. All the other combinations of {H1, H2, H3} were unused and set to the unused state of the fan which also resulted in the fan to be off. The outputs would occur at the rising edge of the clock pulse and the asynchronous reset would cause all outputs to be zero.

B) CoolerTest module

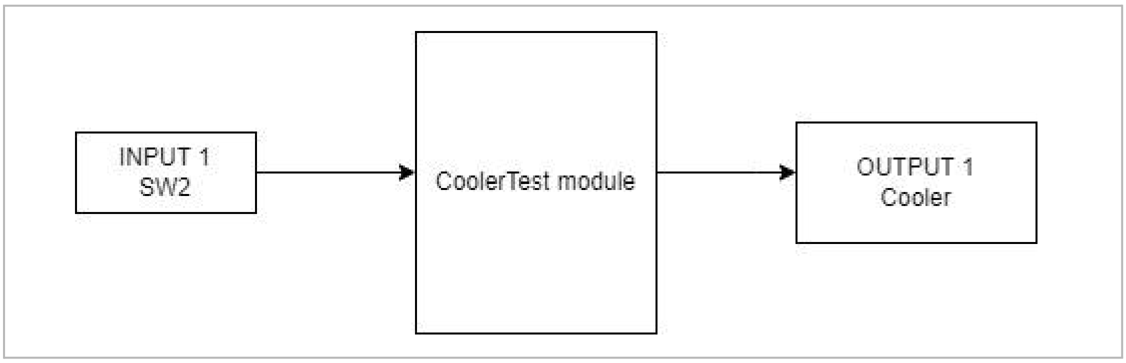


Figure 13 : CoolerTest module [Demonstrated]

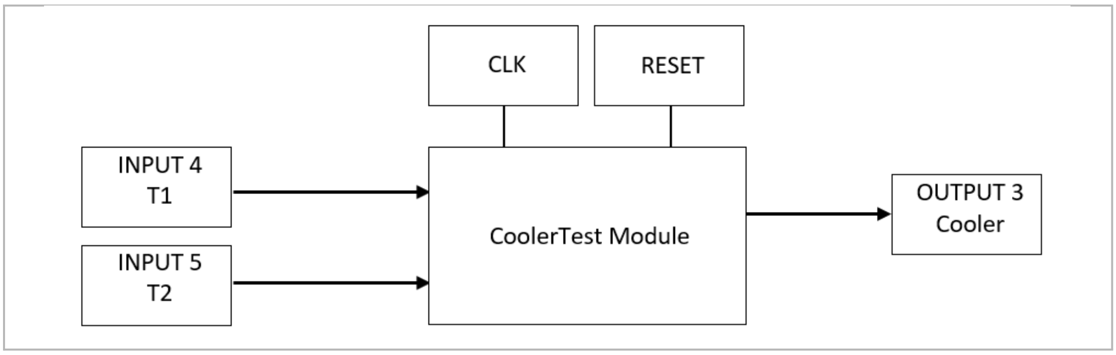


Figure 14 : CoolerTest module [Initial Design]

Figure 13 above shows the block diagram of the demonstrated CoolerTest module. It has only one switch input SW2 and one output Cooler which was connected to the motor driver which operated the peltier cooler. The motor driver was needed for the cooler as it required a high voltage to operate, which the DE1-SoC board could not provide. The input level is the same as the output. When the input was HIGH, it represented a high temperature level which would turn on the peltier cooler, which would lower the temperature. When the input was LOW it represented an acceptable temperature level, the peltier cooler would turn off and the current temperature would be maintained.

Figure 14 shows the block diagram of the initial design of the CoolerTest module. With a similar concept in the Motortest module, the inputs T1 and T2 in the

CoolerTest module were to arrive from the Arduino board from the DHT22 sensor. For testing purposes two temperature ranges were identified and assigned to two different states of {T1, T2}. With reference to the Arduino code in Section 3 of the Appendix; {T1, T2} = 00 represented an acceptable temperature range of 40°C and below which would cause the cooler to be in the off state, {T1, T2} = 11 represented high temperature level of above 40°C which would incur the peltier cooler to turn on. All the other combinations of {T1, T2} were unused and set to the off of the cooler. The outputs would occur at the rising edge of the clock pulse and the asynchronous reset would cause all outputs to be zero.

5. ANALYSIS AND RESULTS

A. *Demonstrated Results*

The DHT22 (temperature and humidity sensor) was unable to be connected to the DE1-SoC board. It was to be connected to an Arduino board then interfaced with the DE1 board however it was not successful due to issues related to the synchronisation of the three components. In order to represent the changes in temperature and humidity that would have come from the sensor, switches were used as the inputs. The inputs from the switches were used to send output signals to the GPIO pins. The components were connected to a motor driver, which enabled the functionality of the dual-way fan and cooler. Below shows the truth tables of the switch inputs and the outputs produced.

INPUT			OUTPUT	
Humidity Level	SW0 [MD1]	SW1 [MD1]	Dual-way Fan	Description
Acceptable	0	0	0	OFF
Low	0	1	1	FAN IN
High	1	0	1	FAN OUT
Unused state	1	0	0	OFF

Table 1: Truth table of fan operation

When the humidity level is too low, the dual-way fan is meant to bring the outside air into the bag, thus causing an increase in humidity levels inside the bag. This was represented by inputs {SW0, SW1}= 01, which resulted in the motor of the fan to rotate in one direction.

When the humidity level is too high, the dual-way fan is meant to bring the air from inside the bag to the outside resulting in the decrease in humidity levels inside the bag. This was represented by inputs {SW0, SW1}= 10, which resulted in the motor of the fan to rotate in the opposite direction.

When the humidity levels are at an acceptable level, the fan is off. This is represented by the inputs {SW0, SW1}= 00. The inputs {SW0, SW1}= 11 are unused, so they were made to be a 'default' whereby the fan would be off.

INPUT		OUTPUT	
Temperature Level	SW2	Cooler	Description
Acceptable	0	0	OFF
High	1	1	ON

Table 2: Truth table of cooler operation

When the temperature increases past a certain point inside the bag, the cooler turns on and vice versa. To represent a temperature level that is above the threshold, the input SW2 = 1 was used. This caused the cooler to turn on. In order to represent an acceptable temperature level, the input SW2 = 0 was used. This caused the cooler to turn off.

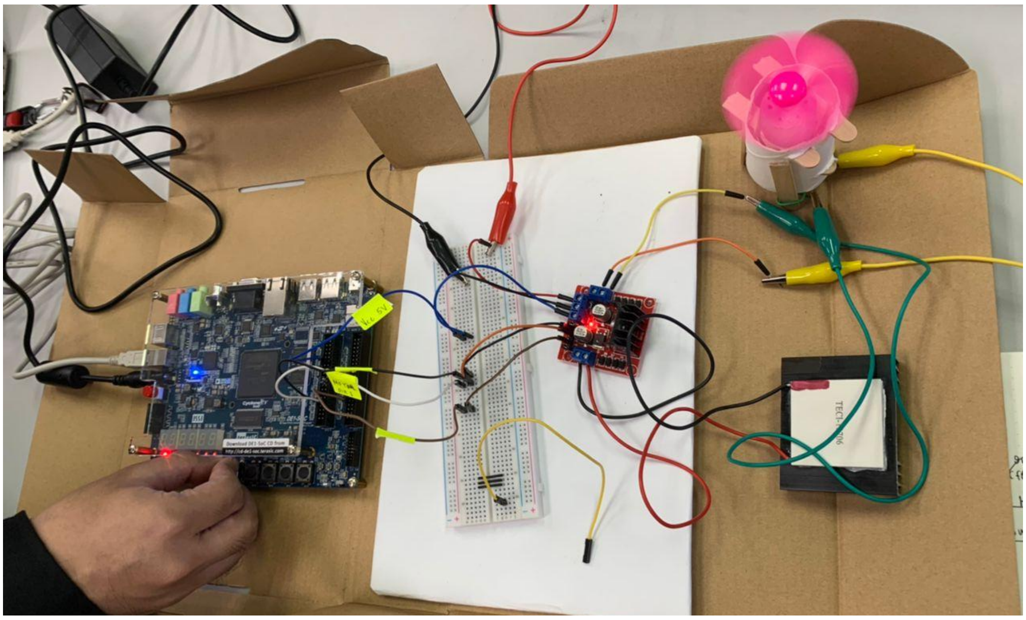


Figure 15 : Demonstrated circuit in operation

	tatu	From	To	Assignment Name	Value	Enabled	Entity	Comment
1	✔		in sw0	Location	PIN_AB12	Yes		Humidity Switch
2	✔		in sw1	Location	PIN_AC12	Yes		Humidity Switch
3	✔		in sw2	Location	PIN_AF9	Yes		Temperature Switch
4	✔		out MD1	Location	PIN_AE18	Yes		Connected to GPIO
5	✔		out MD2	Location	PIN_AE19	Yes		Connected to GPIO
6	✔		out cooler	Location	PIN_AK22	Yes		Connected to GPIO
7	✔		out LED	Location	PIN_Y21	Yes		Test Indicator

Figure 16 : Pin assignments of demonstrated circuit

Entity:Instance
Cyclone V: 5CSEMA5F31C6
pacqed
Motortest:module1
CoolerTest:module2

Figure 17 : Module hierarchy demonstrated circuit

Flow Status	Successful - Tue Feb 01 12:57:09 2022
Quartus Prime Version	20.1.1 Build 720 11/11/2020 SJ Lite Edition
Revision Name	pacqed
Top-level Entity Name	pacqed
Family	Cyclone V
Device	5CSEMA5F31C6
Timing Models	Final
Logic utilization (in ALMs)	2 / 32,070 (< 1 %)
Total registers	0
Total pins	7 / 457 (2 %)
Total virtual pins	0

Figure 18 : Resource usage of Demonstrated Design

B. Initial Design Simulation Results

The demonstrated circuit had a combinational design and the inputs had to be manually asserted onto the board. The initial design had a sequential design whereby the circuit was clocked and the inputs were received from the DHT22 sensor.

The sensor was connected to an Arduino board whereby the data from the sensor could have been transmitted to the DE1-SoC board. Three outputs from the Arduino; H1, H2 and H3 represented the humidity level. Two more outputs from the Arduino; T1 and T2 represented the temperature level. These 5 bits were to be input into the DE1 board via the GPIO pins. As explained above, the humidity levels and temperature would cause the dual-way fan and cooler to operate respectively.

INPUT				Motor Driver Input		OUTPUT	
Humidity Level	HI	H2	H3	MD1	MD2	Dual-way Fan	Description
Unused state	0	0	0	1	1	0	OFF
Low	0	0	1	0	1	1	FAN IN
Acceptable	0	1	0	0	0	0	OFF
Unused state	0	1	1	1	1	0	OFF
High	1	0	0	1	0	1	FAN OUT
Unused state	1	0	1	1	1	0	OFF
Unused state	1	1	0	1	1	0	OFF
Unused state	1	1	1	1	1	0	OFF

Table 3: Expected ruth table of fan operation

It is observed that only three states are utilised to represent humidity levels. The others are left unused. The one-hot encoding method was used.

INPUT			OUTPUT	
Temperature Level	T1	T2	Cooler	Description
Acceptable	0	0	0	OFF
Unused state	0	1	0	OFF
Unused state	1	0	0	OFF
HIgh	1	1	1	ON

Table 4: Expected truth table of cooler operation

Flow Status	Successful - Tue Feb 01 11:29:16 2022
Quartus Prime Version	20.1.1 Build 720 11/11/2020 SJ Lite Edition
Revision Name	pacqed
Top-level Entity Name	pacqed
Family	Cyclone V
Device	5CSEMA5F31C6
Timing Models	Final
Logic utilization (in ALMs)	2 / 32,070 (< 1 %)
Total registers	3
Total pins	10 / 457 (2 %)
Total virtual pins	0

Figure 19 : Resource usage of Initial Design

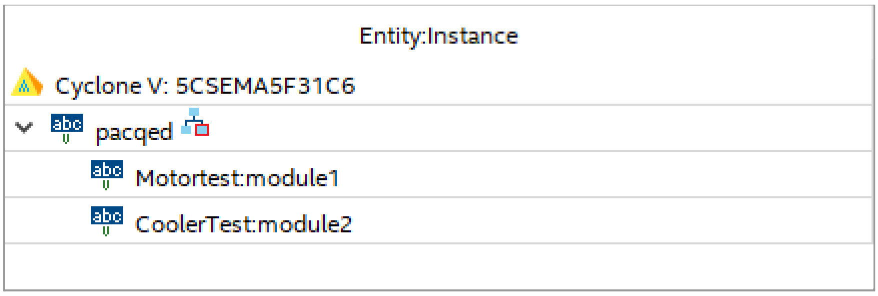


Figure 20 : Module hierarchy initial design circuit

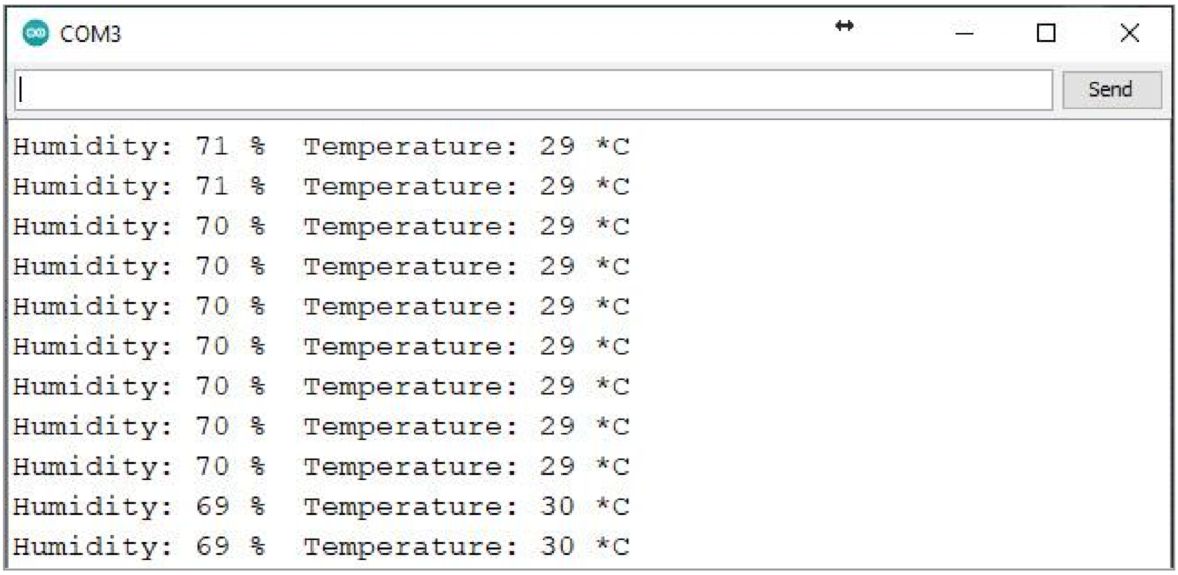


Figure 21 : Temperature and Humidity readings obtained from DHT22 sensor using Arduino

Since the Arduino was unable to be interfaced with the DE1-SoC board, the functional simulation was carried out on Quartus Prime.

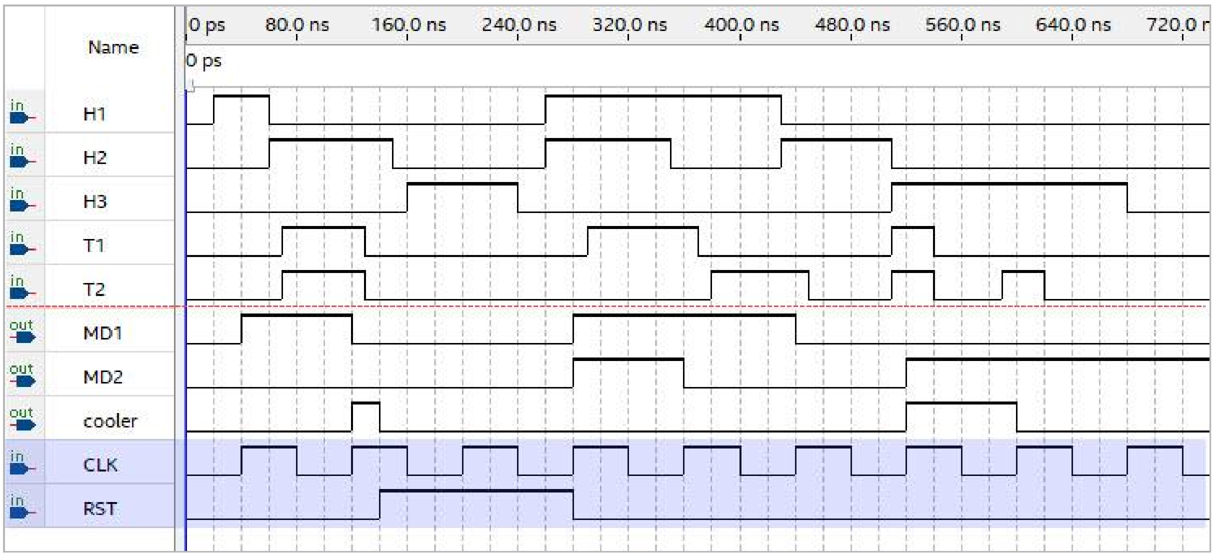


Figure 22 : Functional Simulation using Quartus Prime

The functional simulation shows the expected results as obtained from the truth tables above. The outputs are asserted at the rising edge of the clock pulse. During the asynchronous reset, all the outputs are set to zero regardless of the input.

When the input {H1, H2, H3} = 001 arrives from the Arduino, it represents a low humidity level inside of the bag. This results in the output of {MD1, MD2} = 01, which is fed into the motor driver as inputs and turns on the dual-way fan to the 'FAN IN' direction whereby the outside air is let into the bag to increase the humidity levels. When the input is {H1, H2, H3} = 010, it represents an acceptable humidity level inside of the bag. This results in the output of {MD1, MD2} = 00, which is fed into the motor driver as inputs and the dual-way fan is in its off state to maintain the current humidity levels inside of the bag. When the input {H1, H2, H3} = 100 arrives from the Arduino, it represents a high humidity level inside of the bag. This results in the output of {MD1, MD2} = 10, which is fed into the motor driver as inputs and turns on the dual-way fan to the 'FAN OUT' direction whereby the air inside the bag is let out to reduce humidity levels. All the other input states of {H1, H2, H3} are unused which result in the output of {MD1, MD2} = 11. This also keeps the dual-way fan in the off state.

Similarly for the operation of the cooler, the inputs {T1, T2} are meant to arrive from the Arduino. When the inputs are {T1, T2} = 00, it represents an acceptable temperature range which gives the output Cooler = 0. This keeps the cooler off, so no changes to temperature are made inside the bag. When the inputs are {T1, T2} = 11, it represents a temperature that is above the acceptable threshold which gives the output Cooler = 1, which turns it on cooling the inside of the bag.

6. CONCLUSION

The overall project goal was accomplished whereby a product was designed to fit the given theme of 'Smart Cities: Digital Solutions for a More Livable Future'. The designed product was a handbag, 'PACQED'; that could monitor and regulate humidity and temperature levels to ensure optimal storage conditions of contents inside the bag. Certain design features were identified to be implemented on the DE1-SoC board. Two-thirds of the initial objectives were completed, whereby two of the output functionalities were implemented onto the DE1-SoC board; the dual-way fan and the peltier cooler. They were run simultaneously while being controlled by the DE1 board.

However, the input functionality of obtaining data from the DHT22 temperature and humidity sensor was unable to be done. The sensor needed to be connected with the Arduino UNO board, which would then have to be interfaced with the DE1 board. Readings for temperature and humidity were obtained from the Arduino board, as shown in Figure 21. However, this data could not be sent to the DE1 board due to synchronisation issues between the three components. The DHT22 sensor had its own timing pattern which needed to be synchronised with the Arduino and the DE1 board. There was difficulty in 'matching' the different clocks as the DE1 board has a clock of 50MHz while the DHT22 sensor operates at 0.5Hz. An alternative method was used to replace the sensor inputs with manual switch inputs on the DE1 board to overcome the issue. However, the functionality of PACQED was still able to be demonstrated as expected. The DHT22 sensor was selected due to availability. It could have been replaced with simpler and separate sensors in order to achieve the initial goals. This was considered however, it was unable to be done due to time constraints and accessibility.

A shortfall of the design was highlighted during the pitching session whereby the 'investor' pointed out that the 'PACQED' handbag would be inconvenient to use due to the resulting condensation from using the peltier cooler. Also, they recommended changing the product from a handbag to a smaller separate module or pouch. In the future, the product could be improved by adding new functionalities that could combat the condensation issue and down-sizing the product, which would allow the expansion of the target market and make the product more 'user-friendly'.

In addition, some difficulties were faced during the project period. Some of the difficulties faced were:

- Finding suitable components - to ensure that the components are able to be implemented onto the DE1-SoC board
- Requiring a separate power supply for the external components to function due to their high power usage.
- Damage of components due to faulty or incorrect connections.

Many new skills and experiences were gained throughout the project period. Technical skills were developed; the group members learned the design process of a product, the implementation of different inputs and outputs onto the DE1-SoC board using Verilog entry, carrying out various research methods and learning about the different aspects considered when starting a start-up company. The characteristics of a successful product were learnt; the design and functionality of the product, and the market demand. Furthermore, the soft skills of the group members were also developed and tested; teamwork, communication and problem-solving were the main skills used for the success of this project.

Therefore, given the above reasoning, this design project is definitely beneficial in developing academic strength and industrial skills. A suggestion to improve the project and its potential benefits would be to have a compulsory discussion with an industry professional about the proposed designs. The opinions and feedback may be used to further improve the designed products. This could potentially aid in reducing the shortfalls encountered later in the project as well as giving students more industrial exposure and deeper understanding and appreciation of the product design process.

7. APPENDIX

Section 1: Verilog Code for Demonstrated Results

TOP LEVEL

```
module pacqed ( input sw0 ,sw1,sw2,
                 output MD1, MD2, cooler, LED);

 Motortest module1 (.switch0(sw0) ,.switch1(sw1),.out1(MD1),.out2(MD2));

 CoolerTest module2 (.switch0(sw2), .out0(cooler), .LED9(LED));

endmodule
```

DUAL-WAY FAN MODULE [Motortest]

```
module Motortest (switch0, switch1, out1, out2);

 input switch0, switch1;
 output wire out1, out2;

 reg dir1, dir2;

 always @(*)
      begin
            case({switch0, switch1})

                  2'b00 : begin
                                dir1 <= 0;
                                dir2 <= 0;
                                end

                  2'b01 : begin
                                dir1 <= 0;
                          dir2 <= 1;
                                end

                  2'b10 : begin
                          dir1 <= 1;
                       dir2 <= 0;
                                end

                  2'b11 : begin
                          dir1 <= 0;
                          dir2 <= 0;
                                end

            endcase
      end

 assign out1 = dir1;
 assign out2 = dir2;

endmodule
```

COOLER MODULE [CoolerTest]

```
module CoolerTest (switch0, out0, LED9);

 input switch0;

 output wire out0, LED9;

 reg cooler, L9;


 always @(*)
      begin
            case(switch0)

                  1'b0 :begin             // Cooler ON
                        cooler <= 0;
                        L9 <= 0;
                        end

                  1'b1 : begin            // Cooler OFF
                         cooler <= 1;
                         L9 <= 1;
                         end
            endcase
      end

 assign out0 = cooler;
 assign LED9 = L9;                  //LED INDICATOR

endmodule
```

Section 2: Verilog Code for Initial Design

VERILOG CODE
TOP LEVEL

```
module pacqed ( input H1, H2, H3, T1, T2, CLK, RST,
                output MD1, MD2, cooler);

 Motortest module1 (.IN1(H1) ,.IN2(H2), .IN3(H3), .CLOCK(CLK),
.RESET(RST), .out1(MD1),.out2(MD2));

 CoolerTest module2 (.IN4(T1) ,.IN5(T2), .CLOCK(CLK), .RESET(RST),
.out0(cooler));

endmodule
```

DUAL-WAY FAN MODULE [Motortest]

```
module Motortest (IN1, IN2, IN3, out1, out2, CLOCK, RESET);

 input IN1, IN2, IN3, CLOCK, RESET;

 output wire out1, out2;
```

```
reg dir1, dir2;

always @(posedge CLOCK, posedge RESET)
     begin

     if (RESET == 1)
           begin
                 dir1 <= 0;
                 dir2 <= 0;
           end

     else
           begin
                 case({IN1, IN2, IN3})

                       3'b000 : begin    // unused state [FAN OFF]
                                   dir1 <= 1;
                                   dir2 <= 1;
                                end

                       3'b001 : begin          // Low Humidity [FAN IN]
                                   dir1 <= 0;
                                   dir2 <= 1;
                                end

                       3'b010 : begin    // Acceptable Humidity [FAN OFF]
                                   dir1 <= 0;
                                   dir2 <= 0;
                                end

                       3'b011 : begin    // unused state [FAN OFF]
                                   dir1 <= 1;
                                   dir2 <= 1;
                                   end

                       3'b100 : begin    // High Humidity [FAN OUT]
                                   dir1 <= 1;
                                   dir2 <= 0;
                                end

                       3'b101 : begin    // unused state [FAN OFF]
                                   dir1 <= 1;
                                   dir2 <= 1;
                                end

                       3'b110 : begin    // unused state [FAN OFF]
                                   dir1 <= 1;
                                   dir2 <= 1;
                                end

                       3'b111 : begin    // unused state [FAN OFF]
                                   dir1 <= 1;
                                   dir2 <= 1;
                                end
                 endcase
           end
     end
assign out1 = dir1;
assign out2 = dir2;

endmodule
```

COOLER MODULE [CoolerTest]

```
module CoolerTest (IN4, IN5, out0, CLOCK, RESET);

 input IN4, IN5, CLOCK, RESET;
 output wire out0;
 reg cooler;


 always @(posedge CLOCK, posedge RESET)
      begin

            if (RESET == 1)
                  cooler<=0;

            else
                  begin
                        case({IN4, IN5})

                              2'b00 : cooler <= 0; // cooler OFF

                              2'b01 : cooler <= 0; // unused state

                              2'b10 : cooler <= 0; // unused state

                              2'b11 : cooler <= 1; // cooler ON

                        endcase
                  end

      end

 assign out0 = cooler;

endmodule
```

Section 3: Arduino Code

```
#include "DHT.h"          // DHT 22 Arduino Library
#define DHTPIN 2          //sensor to arduino at PIN 2
#define DHTTYPE DHT22     // DHT 22

DHT dht(DHTPIN, DHTTYPE);

void setup() {

  // Temp out pins
  pinMode(5, OUTPUT);
  pinMode(6, OUTPUT);

  // Humidity out pins
  pinMode(11, OUTPUT);
  pinMode(12, OUTPUT);
  pinMode(13, OUTPUT);
```

```
  pinMode(LED_BUILTIN, OUTPUT);
  Serial.begin(9600);
  Serial.println("DHTxx test!");
  dht.begin();
}

void loop() {
  // Wait a few seconds between measurements.
  delay(2000);

  // Reading temperature or humidity takes about 250 milliseconds!
  // Sensor readings may also be up to 2 seconds 'old'
  int h = dht.readHumidity();
  int t = dht.readTemperature();

  // Check if any reads failed and exit early (to try again).
  if (isnan(h) || isnan(t))
  {
    Serial.println("Failed to read from DHT sensor!");
    return;
  }

  Serial.print("Humidity: ");   Serial.print(h);   Serial.print(" %\t");
  Serial.print("Temperature: ");  Serial.print(t); Serial.print(" *C\n");


 // TEMP OUTPUTS
  if (t<=40)
  {
  digitalWrite(5,LOW);  digitalWrite(6,LOW);
  }
  else if (t>40)
 {
  digitalWrite(5,HIGH); digitalWrite(6,HIGH);
 }


// HUMIDITY OUTPUTS
  if (h<=30)
  {
  digitalWrite(11,LOW);  digitalWrite(12,LOW);  digitalWrite(13,HIGH);
  }
  else if (30<h<80)
 {
  digitalWrite(11,LOW);  digitalWrite(12,HIGH);  digitalWrite(13,LOW);
 }
 else if (h>=80)
 {
  digitalWrite(11,HIGH);  digitalWrite(12,LOW);  digitalWrite(13,LOW);
 }
}
```

8. REFERENCES AND BIBLIOGRAPHIES

[1] "What is a Smart City? – Definition and Examples", *Twi-global.com*, 2022. [Online]. Available:https://www.twi-global.com/technical-knowledge/faqs/what-is-a-smart-city. [Accessed: 01- Feb- 2022].

[2] "The best way to keep cosmetics fresh and prevent infections | Cosmetic Calculator " , Checkcosmetic.net, 2021. [Online].
Available: https://checkcosmetic.net/best-practices-keep-cosmetics-fresh/. [Accessed: 24- Dec- 2021].

[3] W. Konrad, "Mistakes in Storage May Alter Medication (Published 2011)" , Nytimes.com, 2011. [Online]. Available: https://www.nytimes.com /2011/08/16/health/16consumer.html. [Accessed: 24- Dec- 2021].

[4] CTPA. 2021. Preservatives & storage of cosmetic products. [online] Available: https://www.thefactsabout.co.uk/storage-cosmetic-products [Accessed 26 December 2021].

[5] "WHO Technical Report Series", World Health Organisation, Geneva, 2003.

[6] 10 Common Project Risks (Plus How To Analyze and Solve Them) Available: https://www.indeed.com/career-advice/career-development/project-risks