# Café del Campo
## Sparktech

June 24, 2022

## 1 Team members

- Miguel Rodriguez
- Shao-Wen Chang
- Asadujaman Nur
- Vincent Obigwe

## 2 Introduction

According to legend, an Ethiopian Goatherd first discovered the coffee in the 9th century. When he noticed his flock became so active and energetic after consuming some kind of Barry. Later on, he tried the same berry by himself and noticed he could stay awake longer without getting tired because of the presence of caffeine in the berry. When the local monk witnessed this phenomenon they were intrigued and started to boil the bean and drink the water so that they could pray for a longer time even in the night thus the first coffee was born.

Later on due to colonisation in 1640 coffee was imported into Europe by Dutch merchants. After that, various coffees started to emerge. It got huge popularity, it even replaced wine and beer during breakfast since people started to notice drinking a cup of coffee boosted their productivity. Fast forward to the modern era, our life got even busier and therefore to keep our

focus and energy we often drink coffee, from homemade to vending machines. Coffee is one of the most consumed and accessible drink 2nd after water. Moreover, there is no sign of decreasing its popularity. Due to its huge popularity, there are various versions of coffee and various ways to serve.

During the Hardware Engineering Lab we are going to Design and build a prototype of a Coffee vending machine Named "Café del campo" Using FPGA and VHDL. Using "VHDL", we can program our Feature circuits, Logic, and other necessary components from scratch. For the VHDL programming, we are going to use ModelSim software. After programming our features in VHDL, we need to build the necessary circuits so that we can manufacture them. To Design and build our necessary "FPGA" PCB board we are going to Use "Eagle". To realize our project these two methods are crucial because of the reason above. We are also going to implement some other methods like "block diagram, Agile project management, various technologies, VHDL Implementation, PCB Design" etc. which we will discuss in detail later on in the paper.

# 3    Concept Description

A Coffee machine that provides coffees with high quality coffee. According to the type of coffee, the ratio of water, coffee powder, milk or sugar is adjusted. To acquire the desired coffee, press a number for type and another for whether sugar is added. The output will then be water, coffee powder, milk and sugar if selected.

In Figure **??** we have a guide on how are the different combinations of coffee, streamed milk, foamed milk, water and chocolate, depending on the type of coffee that the user can possibly select

The block diagram of the coffee machine is described on Figure **??**. There will be two inputs at the beginning of the project: coffee number and sugar selection, there will be a clock or timer that will determinate the amount of time that the different ingredients will be pouring into the cup and on the right side four initial outputs, coffee, water, milk and sugar. Depending on the time and the complexity of the task there are two extra components that can be added, the cash input and the cash change.

Figure 1: Espresso guide

# 4 Project / Team management

In carrying out our project, we consider project management as an important aspect that helps us in realizing the goal and objective faster. Project management benign the application of various methods, skill knowledge and experience to achieve a detailed object that has already been set and marked as the project acceptance benchmark. It is also very important that time and budget are considered. As students we embarked on this project with the focus on managing time and delivering the best result given the limited amount of time that and the no budget we deal with. Having this in mind, we decided to approach the whole project using the agile project management [?].

Agile project management is known as an iterative means of delivering a project throughout a given life cycle. There life cycle are made up of several
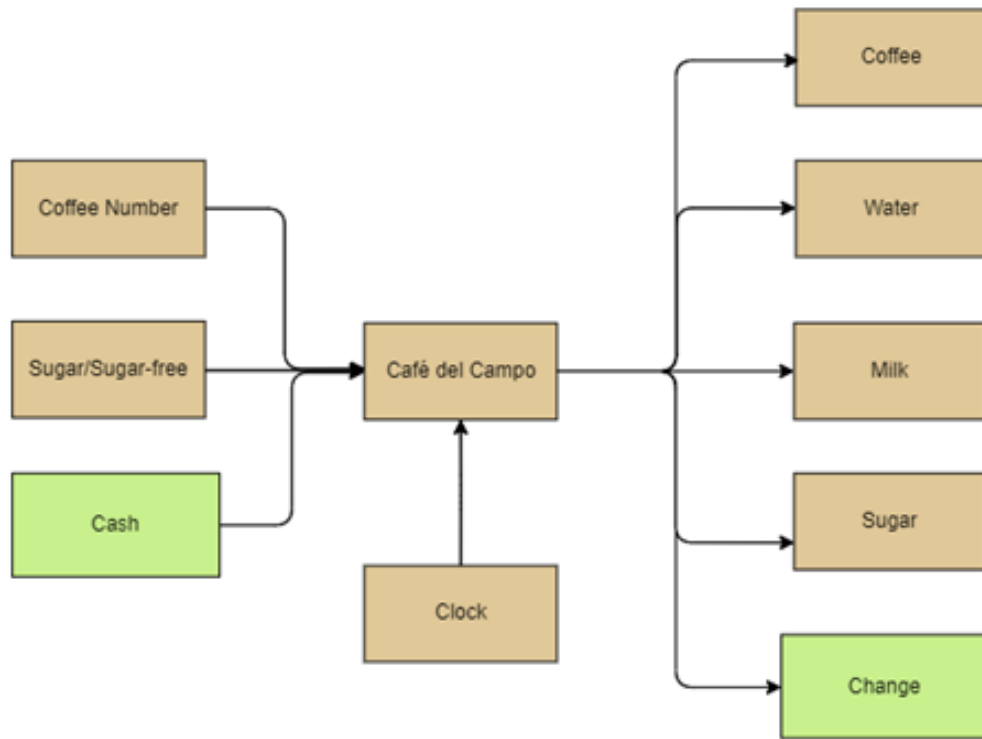
Figure 2: Espresso guide

iterations that are all geared towards the completion of the project. This leads to us having weekly meetings to analyze tasks and evaluate progress. With Agile project management, continuous improvement and development is the goal, to enable the project to get better on further iterations. While using Agile project management, we made use of Kanban framework to be able to achieve our project [?].

Kanban is a framework on agile project management that is deals with growth changes and the need for continuous process improvement. The core practices involved in using Kanban are:

1. Visualization of tasks in a board like manner with tools such as excel.
2. Reduction of work in project by introducing changes incrementally
3. Management of the flow of to-dos
4. Defining processes and tasks clearly
5. Enhancing the process of feedback to enable improvement of the sys-

tem.

6. Improving workflow all together.

To be able to properly visualize our tasks and clearly see what each member of the group has to do, we made use of GitHub projects. GitHub projects can be seen as a customizable spreadsheet that helps us integrate tasks with GitHub in our repository. It empowers more customization by enabling filtering; sorting, grouping and working with GitHUb pull requests. It further looks static with the addition of colors to determine the various stage each task is [**?**].

# 5 Technologies

## 5.1 VHDL

One of the technologies that we will use for this project is VHDL a hardware description language [**?**]. This language can model the behaviour of a digital system in different levels of abstraction [**?**]

## 5.2 FPGA

Field programmable gate arrays popularly known as FPGA are increasingly used to power electronic devices. A complex control algorithm can easily be implemented using FPGA for any given electronic system. Xilinx-ISE is a platform that is used for a variety of reasons depending on one's usecase. However, it is predominantly used in relation to FPGA development. It can be used to write VHDL code and feed it into the FPGA [**?**]. Xilinx-ISE is been used by us for the development of FPGA.

## 5.3 EAGLE

Eagle: According to [**?**] Eagle is an electronic design automation application with schematic capture, printed circuit board layout, auto-router, and Computer-Aided Manufacturing features. EAGLE stands for Easily Applicable Graphical Layout Editor which is developed by CAD-Soft Computer GmbH. The company was acquired by Autodesk Inc in 2016. This is one of the most used software because of its basic free license which one can use for

its study and personal purposes. In this course, we were advised to use eagle to represent our project which will be explained later on in PCB Design.

# 6  Implementation

## 6.1  VHDL

The VHDL code in out project is divided in three parts. The first one will take the input of the user, input is given by 6 options, Espresso, Americano, Cappucino, Late, Macchiato and sugar. Moreover, it takes input of a clock to synchronize the system. The first module will take the user input and will organize it in a simple four-bit vector and will output it. This output will be the input of the second module. Figure ??.

```vhdl
1    library ieee;
2    use ieee.std_logic_1164.all;
3
4    entity dummy_coffee_Machine is
5        port (UINPUT : in std_logic_vector(5 downto 0);
6              CLK : in std_logic;
7              ST1_OUT : out std_logic_vector (3 downto 0));
8
9    end entity;
10
11   architecture INPUT of dummy_coffee_Machine is
12
13   begin
14       process(UINPUT, CLK) is
15           begin
16           if rising_edge (CLK) then
17
18               case UINPUT is
19               when "100000" => ST1_OUT <= "0001";
20               when "010000" => ST1_OUT <= "0010";
21               when "001000" => ST1_OUT <= "0011";
22               when "000100" => ST1_OUT <= "0100";
23               when "000010" => ST1_OUT <= "0101";
24               when "100001" => ST1_OUT <= "0110";
25               when "010001" => ST1_OUT <= "0111";
26               when "001001" => ST1_OUT <= "1000";
27               when "000101" => ST1_OUT <= "1001";
28               when "000011" => ST1_OUT <= "1010";
29               when others => ST1_OUT <= "0000";
30               end case;
31           end if;
32
33           end process;
34
35   end INPUT;
36
```

Figure 3: First VHDL module

The second module will take the output of the first module as an input, and will output five four-bit vectors, this vectors will represent the signal with the units that have to be poured for each case, giving a zero as an output when there is not unit that have to be poured, or up to 8, for example the units for the Latte. Figure ??

```vhdl
2    use ieee.std_logic_1164.all;
3
4    entity Coffe_S2 is
5        port (ST1_OUT : in std_logic_vector(3 downto 0);
6              COFFEE: out std_logic_vector (3 downto 0);
7              SUGAR: out std_logic_vector (3 downto 0);
8              MILK: out std_logic_vector (3 downto 0);
9              WATER: out std_logic_vector (3 downto 0);
10             FOAM: out std_logic_vector (3 downto 0));
11
12   end entity;
13
14   architecture Mix of Coffe_S2 is
15
16   begin
17       process(ST1_OUT)
18       begin
19       case ST1_OUT is
20           when "0001" =>
21                   COFFEE <= "0001";
22                   MILK <= "0000";
23                   WATER <= "0000";
24                   SUGAR <= "0000";
25                   FOAM <= "0000";
26
27           when "0010" =>
28                   COFFEE <= "0010";
29                   MILK <= "0000";
30                   WATER <= "0011";
31                   SUGAR <= "0000";
32                   FOAM <= "0000";
33
34           when "0011" =>
35                   COFFEE <= "0010";
36                   MILK <= "0010";
37                   WATER <= "0000";
38                   SUGAR <= "0000";
39                   FOAM <= "0010";
40
41           when "0100" =>
42                   COFFEE <= "0010";
43                   MILK <= "1000";
44                   WATER <= "0000";
45                   SUGAR <= "0000";
46                   FOAM <= "0001";
47
48           when "0101" =>
49                   COFFEE <= "0010";
```

Figure 4: Second VHDL module

The third module combine the two previous models in one simple model. This module is the global representation of the coffee machine. Figure **??**



```
Ln#
 1      library ieee;
 2      use ieee.std_logic_1164.all;
 3
 4    ⊟ entity COFFEE_M_W is
 5    ⊟     port (UINPUT : in std_logic_vector(5 downto 0);
 6                 COFFEE: out std_logic_vector (3 downto 0);
 7                 SUGAR: out std_logic_vector (3 downto 0);
 8                 MILK: out std_logic_vector (3 downto 0);
 9                 WATER: out std_logic_vector (3 downto 0);
10                 FOAM: out std_logic_vector (3 downto 0));
11
12      end entity;
13
14    ⊟ architecture C2TB of COFFEE_M_W is
15
16    ⊟     component dummy_coffee_Machine
17    ⊟     port (UINPUT : in std_logic_vector(5 downto 0);
18                 ST1_OUT : out std_logic_vector (3 downto 0));
19          end component;
20
21          signal ST1_OUT: std_logic_vector(3 downto 0);
22
23    ⊟     component Coffe_S2
24    ⊟     port (ST1_OUT : in std_logic_vector(3 downto 0);
25                 COFFEE: out std_logic_vector (3 downto 0);
26                 SUGAR: out std_logic_vector (3 downto 0);
27                 MILK: out std_logic_vector (3 downto 0);
28                 WATER: out std_logic_vector (3 downto 0);
29                 FOAM: out std_logic_vector (3 downto 0));
30          end component;
31
32          begin
33
34              CM1: dummy_coffee_Machine port map (UINPUT, ST1_OUT);
35              CM2: Coffe_S2 port map (ST1_OUT, COFFEE, SUGAR, MILK, WATER, FOAM);
36
37      end C2TB;
38
```

Figure 5: Third VHDL module

Finally we wrote test benches for each one of the modules to check that the behaviour was as expected. in Figure **??** we have an example of the test bench.

```
Ln#
1     library ieee;
2     use ieee.std_logic_1164.all;
3
4     entity TB_COFFE_WOC is
5
6     end entity;
7
8
9     architecture CTB of TB_COFFE_WOC is
10        component COFFEE_M_W
11        port (UINPUT : in std_logic_vector(5 downto 0);
12               COFFEE: out std_logic_vector (3 downto 0);
13               SUGAR: out std_logic_vector (3 downto 0);
14               MILK: out std_logic_vector (3 downto 0);
15               WATER: out std_logic_vector (3 downto 0);
16               FOAM: out std_logic_vector (3 downto 0));
17        end component;
18        signal UINPUT : std_logic_vector(5 downto 0);
19        signal COFFEE: std_logic_vector (3 downto 0);
20        signal SUGAR: std_logic_vector (3 downto 0);
21        signal MILK: std_logic_vector (3 downto 0);
22        signal WATER: std_logic_vector (3 downto 0);
23        signal FOAM: std_logic_vector (3 downto 0);
24        begin
25
26            UUT: COFFEE_M_W port map (UINPUT, COFFEE, SUGAR, MILK, WATER, FOAM);
27
28            UINPUT <= "100000", "010000" after 100ns, "001000" after 200ns,  "000100" after 300ns,   "000010" after 400ns,
29                "100001" after 500ns, "010001" after 600ns,  "001001" after 700ns,  "000101" after 800ns,   "000011" after 900ns,
30                "110001" after 1000ns, "101010" after 1100ns, "101100" after 1200ns;
31
32
33     end CTB;
```

Figure 6: Test bench

## 6.2 PCB Design

### 6.2.1 Xilinx

The process of design of PCB started with the programming started with ModelSim with the design and simulation of the VHDL code and ends in Xilinx-ISE with the design of the FPGA.

In our scenario the process was as follows:

1. Creation of Project: To start a new project in Xilinx-ISE where you will have to input all the design properties such as family etc. In our case, the following was our unique properties are outlined in the Figure ??.

   These properties are important because they will help in determining how the system would work.

2. Check Syntax: This is done to confirm that the syntax of the VHDL code is correct and it won't throw up an error during the course of synthesis.

9

Figure 7: Xilinx-ISE Design Properties

3. Synthesize-XST: This is important to be able to generate the schematics. After this has been run, two sets of schematics are generated: RTL Schematics (This is the general schematics that can be applied to any FPGA) and Technology Schematics which is applied to a particular device configured, in our case Spartan3.
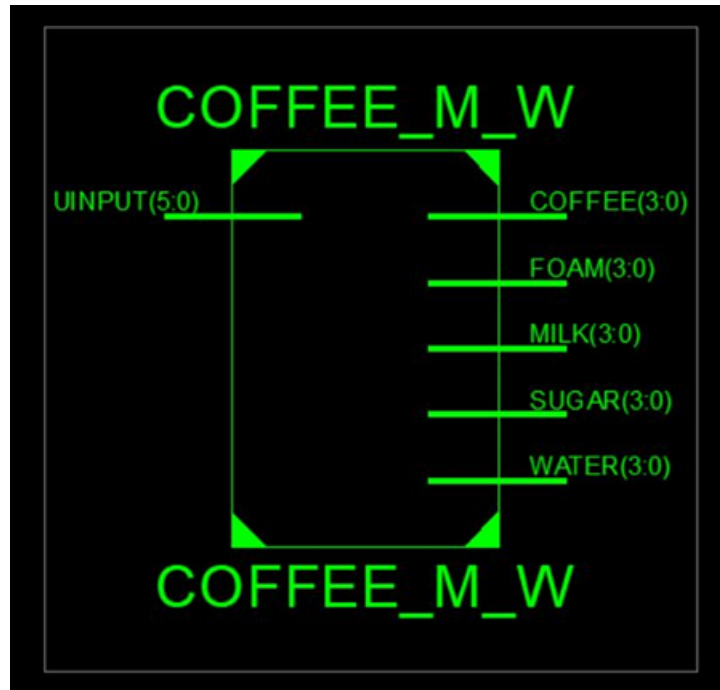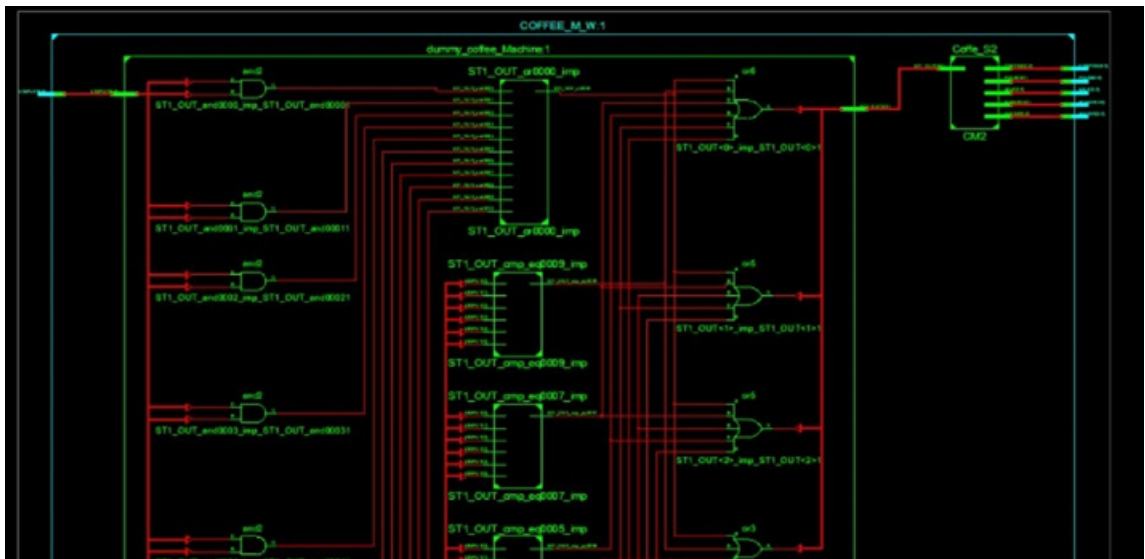
Figure 8: RTL Schematics 1
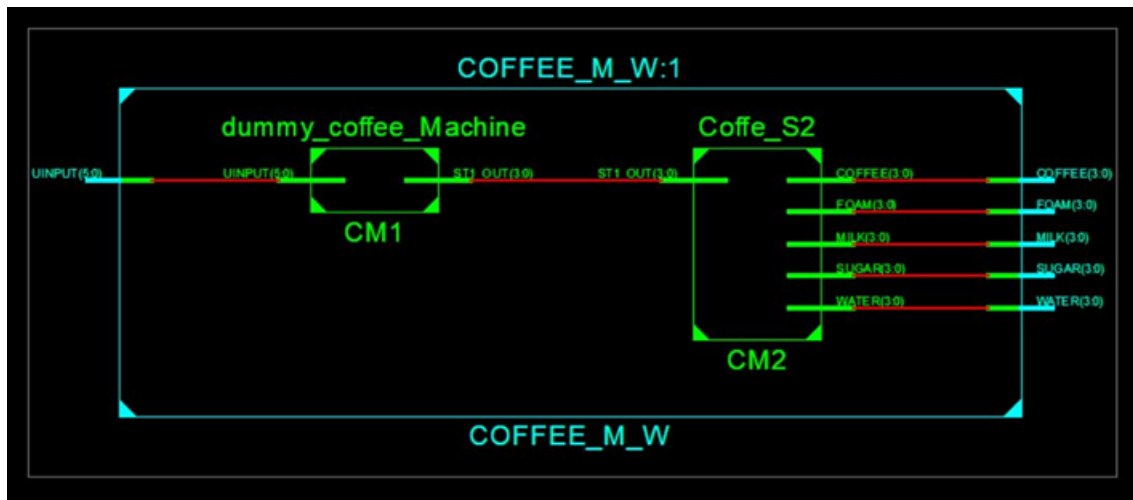


Figure 9: RTL Schematics 2
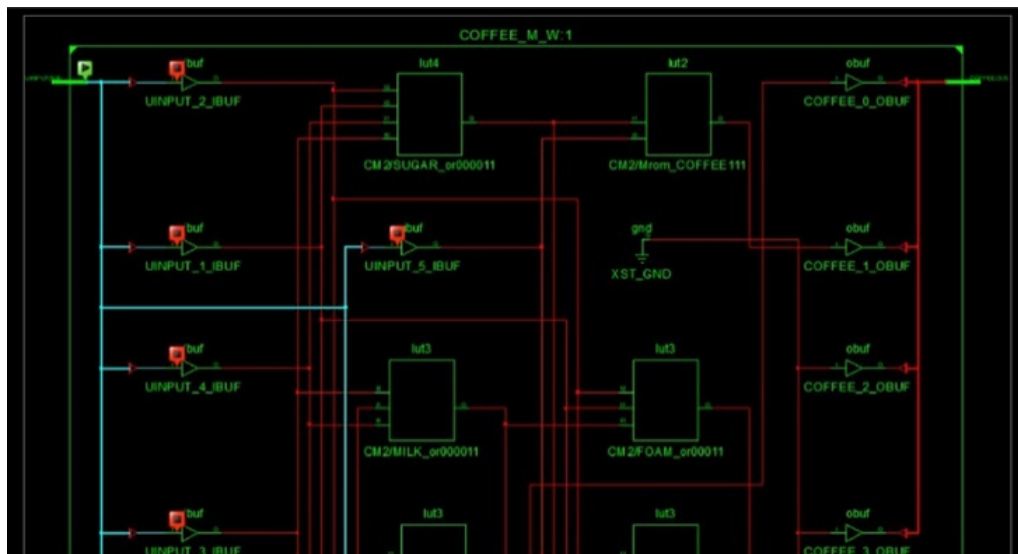
Figure 10: RTL Schematics 3



Figure 11: Technology Schematics

4. Creating the User Constraint file: The user constraint file is used to map the various input and outputs of the device to the pins of the FPGA. The right pin is seen in the documentation of the FPGA. Figure **??**

```
1    # Input (one Input)
2
3    NET "UINPUT<0>"    LOC = "F12" ;
4    NET "UINPUT<1>"    LOC = "G12" ;
5    NET "UINPUT<2>"    LOC = "H14" ;
6    NET "UINPUT<3>"    LOC = "H13" ;
7    NET "UINPUT<4>"    LOC = "J14" ;
8    NET "UINPUT<5>"    LOC = "J13" ;
9
10
11   # Outputs using LED (5 output)
12
13   NET "COFFEE<0>"    LOC = "K12" ;
14   NET "FOAM<0>"  LOC = "P14" ;
15   NET "MILK<0>"  LOC = "L12" ;
16   NET "SUGAR<0>" LOC = "N14" ;
17   NET "WATER<0>" LOC = "P13" ;
18
19
```

Figure 12: User constraint file

5. Implement Design: This was the final step of our FPGA design/ programming, such that it was used to generate the actual design and map the various FPGA pins. The diagram confirms how it will appear with the board and how we can easily interact with it. Figures **??**, **??**, **??** and **??** shows scans from the result generated.
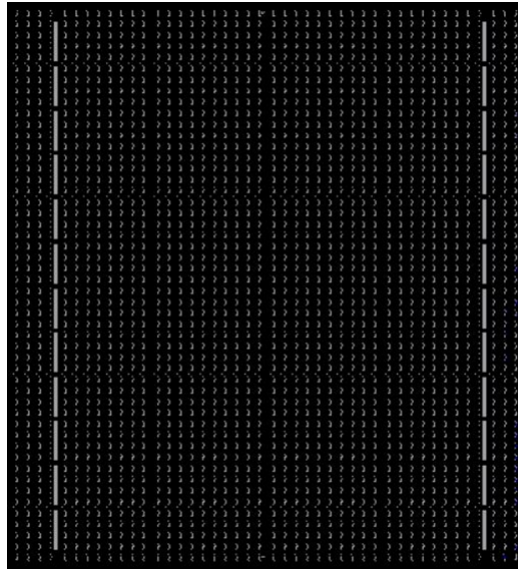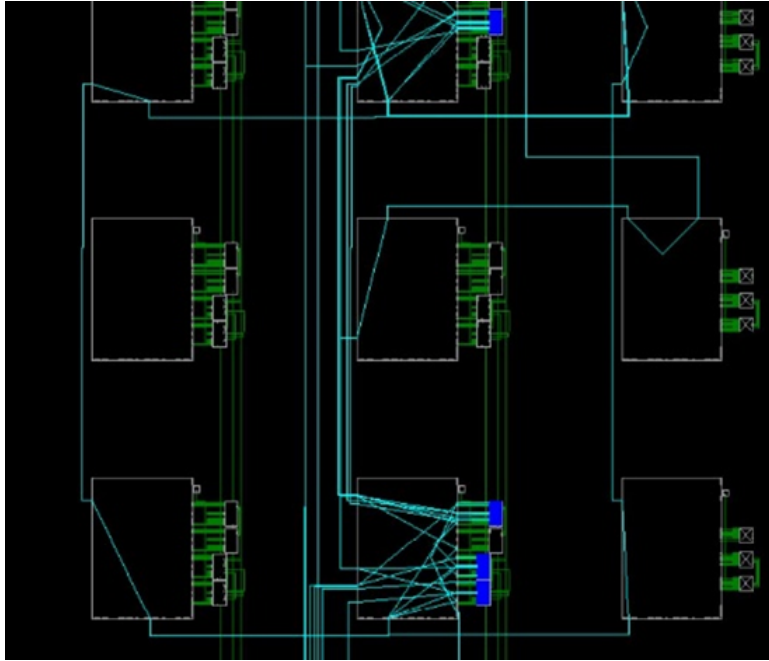


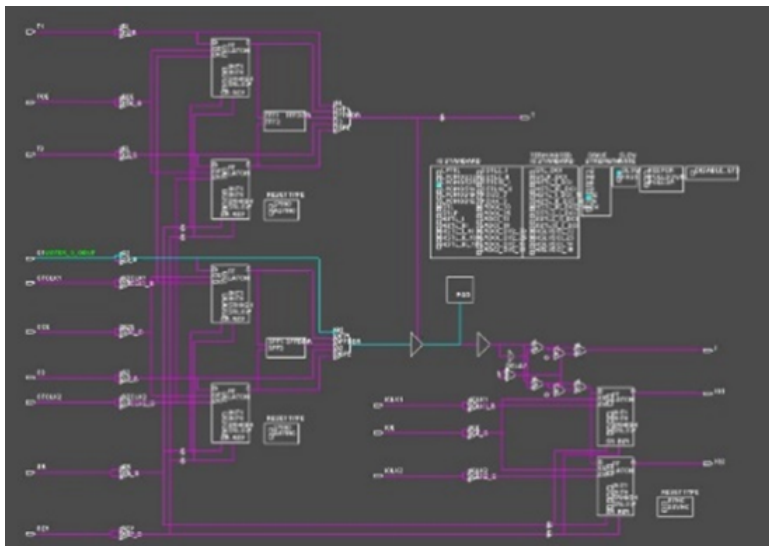Figure 13: Implement design 1

Figure 14: Implement design 2



Figure 15: Implement design 3

| COFFEE_M_W Project Status | | | |
|---|---|---|---|
| Project File: | Coffee_de_campo.xise | Parser Errors: | No Errors |
| Module Name: | COFFEE_M_W | Implementation State: | Placed and Routed |
| Target Device: | xc3s1000-5ft256 | • Errors: | No Errors |
| Product Version: | ISE 14.7 | • Warnings: | 5 Warnings (0 new) |
| Design Goal: | Balanced | • Routing Results: | All Signals Completely Routed |
| Design Strategy: | Xilinx Default (unlocked) | • Timing Constraints: | |
| Environment: | System Settings | • Final Timing Score: | 0 (Timing Report) |

| Device Utilization Summary | | | | | [-] |
|---|---|---|---|---|---|
| Logic Utilization | Used | Available | Utilization | Note(s) | |
| Number of 4 input LUTs | 11 | 15,360 | 1% | | |
| Number of occupied Slices | 6 | 7,680 | 1% | | |
| Number of Slices containing only related logic | 6 | 6 | 100% | | |
| Number of Slices containing unrelated logic | 0 | 6 | 0% | | |
| Total Number of 4 input LUTs | 11 | 15,360 | 1% | | |
| Number of bonded IOBs | 26 | 173 | 15% | | |
| Average Fanout of Non-Clock Nets | 2.63 | | | | |

Figure 16: Design Summary 1

| Performance Summary | | | | [-] |
|---|---|---|---|---|
| Final Timing Score: | 0 (Setup: 0, Hold: 0) | Pinout Data: | Pinout Report | |
| Routing Results: | All Signals Completely Routed | Clock Data: | Clock Report | |
| Timing Constraints: | | | | |

| Detailed Reports | | | | | | [-] |
|---|---|---|---|---|---|---|
| Report Name | Status | Generated | Errors | Warnings | Infos | |
| Synthesis Report | Current | Thu 23. Jun 15:30:22 2022 | 0 | 0 | 0 | |
| Translation Report | Current | Thu 23. Jun 15:37:12 2022 | 0 | 0 | 0 | |
| Map Report | Current | Thu 23. Jun 15:37:16 2022 | 0 | 0 | 2 Infos (0 new) | |
| Place and Route Report | Current | Thu 23. Jun 15:37:21 2022 | 0 | 5 Warnings (0 new) | 2 Infos (0 new) | |
| Power Report | | | | | | |
| Post-PAR Static Timing Report | Current | Thu 23. Jun 15:37:24 2022 | 0 | 0 | 6 Infos (0 new) | |
| Bitgen Report | | | | | | |

| Secondary Reports | | | [-] |
|---|---|---|---|
| Report Name | Status | Generated | |

Figure 17: Design Summary 2

### 6.2.2 Eagle

As discussed before, in this course we were presented an opportunity to design a coffee machine using VHDL (Modelsim), FPGA (Xilings and Eagle). While the rest of the topic goes in depth, the eagle PCB design presents more of an abstract version of the project. Which we will discuss systematically in this section.

From the very beginning right after we open the software we had to create a project. In the project, we had to create a schematic file, which will allow us to create a schematic for the project that can be used later on for layout purpose, Figure **??**.
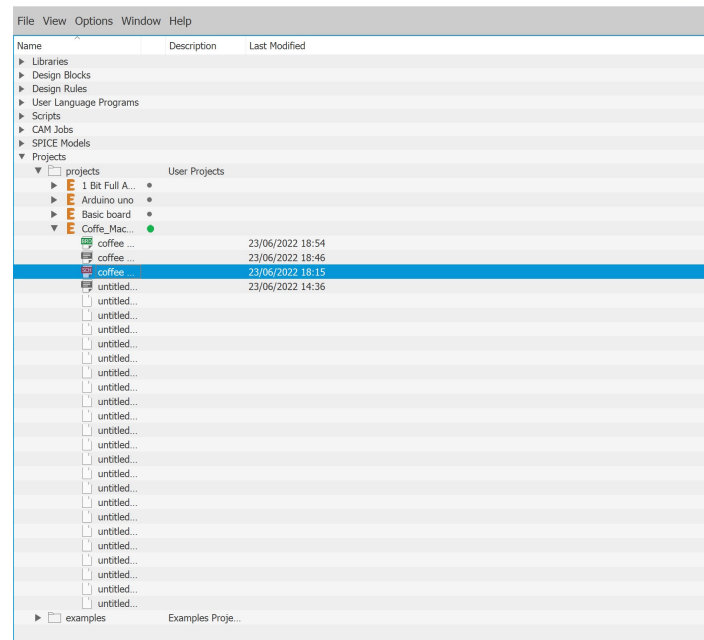


Figure 18: Creating Project and Schematic

Once we created the schematic, we can now add component as necessary. To represent the button designated to the coffee we add a SWITCH-_PUSHBUTTON component From Adafruit library which we were advised to install during our lab period along with SparkFun. Since we have five different kinds of coffee, we added 5 SWITCH_PUSHBUTTON Designated with the name and the number of the drinks like:

1. ESPRESSO

2. AMERICANO

3. CAPPUCCINO

4. LATTE

5. MACCHIATO

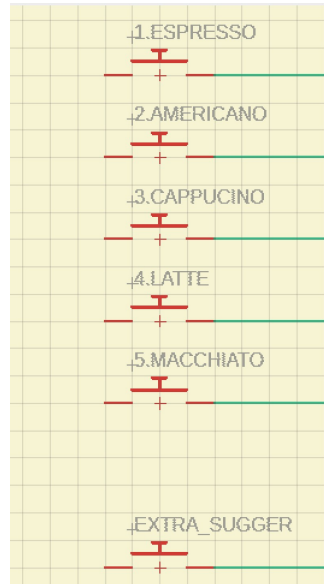and an extra button for adding sugar (6. SUGAR) Figure **??**



Figure 19: Button for coffee

Since we have the necessary input now we need a microcontroller to process the algorithm for making the coffee. Therefor we choose Arduino uno r3 among many. Shown in the Figure **??**
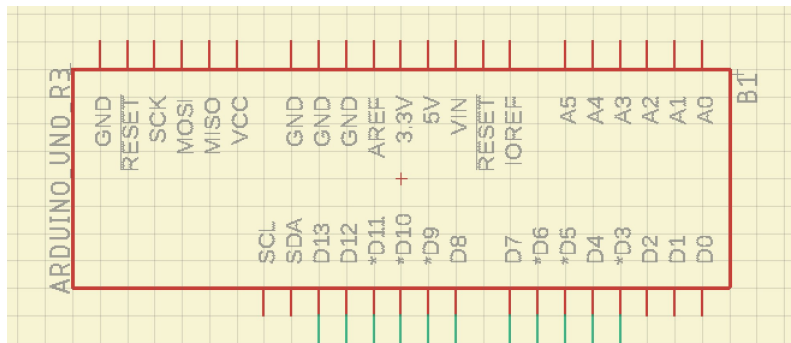


Figure 20: Arduino uno

Now we need the last component to proceed to the next step. The final component will be something that shows the output for our coffee machine. And to show the output we choose LED as our component. After adding the LED and resister to our schematic, we labeled it as our five output (Fig **??**):
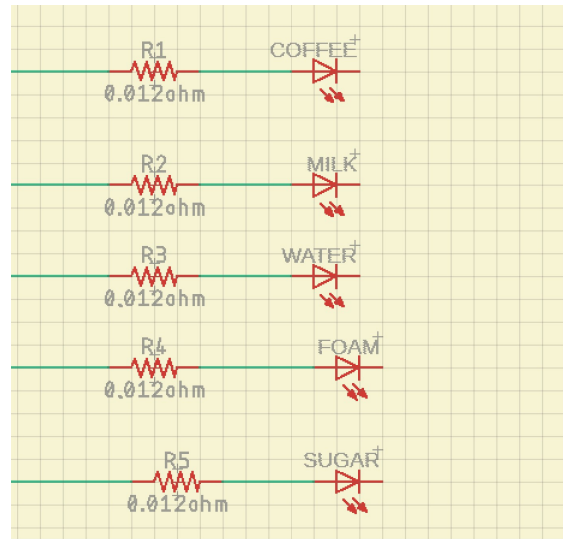
1.COFFEE
2.MILK
3.WATER
4.FOAM
5.SUGAR



Figure 21: Leds and resistors

Since we have all the required components, we can start to wire them together. To wire them we have to use net tool from the tool bar. In addition, for the connection we wire the ports as shown on the table, as well as in the Figure **??**

Input:
1. ESPRESSO to D13
2.AMERICANO to 12
3.CAPPUCINO to D11

4.LATTE to D10
5.MACCHIATO to D9
6. EXTRA_SUGAR to D8

Output:
1.COFFE to D3
2.MILK to D4
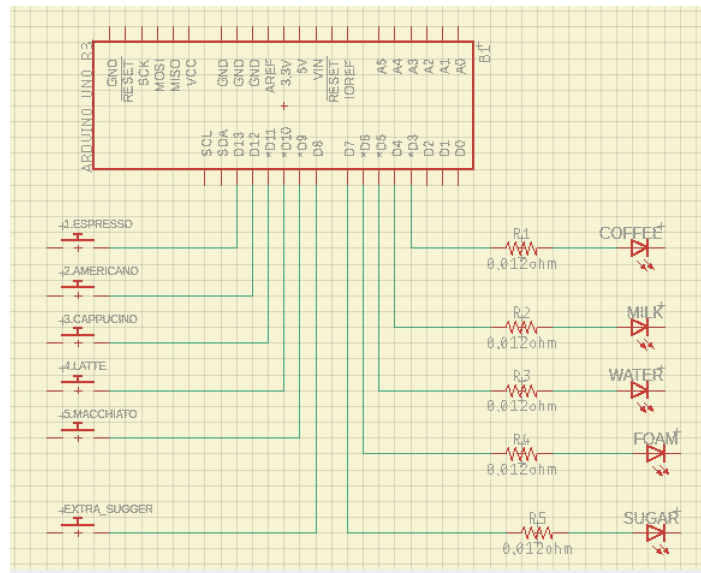3.WATER to D5
4.FOAM to D6
5.SUGAR to D7



Figure 22: System connections

Once we are done with our schematic, we can start the layout process. Since our project is a simple one, we chose single one-sided layer to put out components on the board. We could jump to the layout section. Once we are in the layout window we can start mapping out component such a way that it reduce better prevent cross wiring and clutter. For that we did the component placement manually and for wiring we used auto command. Using auto command, it reduce the workload a lot. Once the process is done then we start to manually correct the wiring as necessary. Some time moving the wires or component and sometime rotating the component, either way

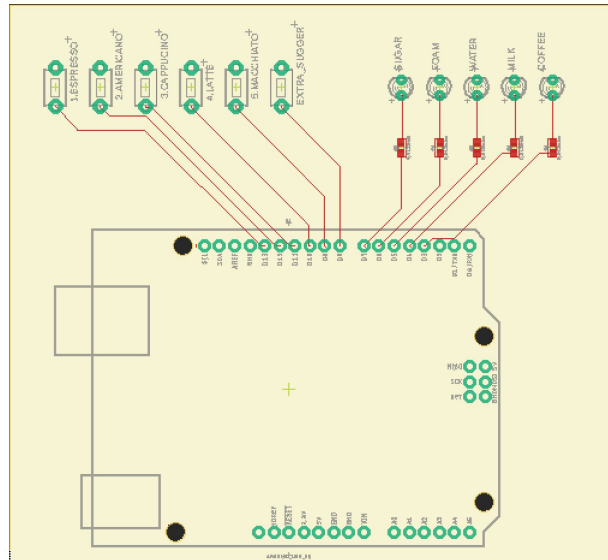which ever provides the best outcome. The Layout can be seen from the Figure **??**.



Figure 23: Layout of Café Del Campo

By following the steps from schematic to layout we were able to achieve the Café del campo using AutoDesk Eagle.

# References

[1] Kanvanize. (n.d.). What Is Agile Project Management? A Comprehensive Guide. Kanban Software for Agile Project Management. https://kanbanize.com/agile/project-management

[2] APM. (n.d.). Agile project management. What Is Agile Project Management? https://www.apm.org.uk/resources/find-a-resource/agile-project-management/

[3] APM. (n.d.-b). What is project management? https://www.apm.org.uk/resources/what-is-project-management/

[4] GitHub. (n.d.). About projects. About Projects. https://docs.github.com/en/issues/trying-out-the-new-projects-experience/about-projects

[5] Palanisamy, R., Boopathi, C. S., Selvakumar, K., and Vijayakumar, K. (2020). Switching pulse generation for DC-DC boost converter using Xilinx-ISE with FPGA processor. International Journal of Electrical and Computer Engineering, 10(2), 1722.

[6] Wikipedia contributors. (2022, February 1). EAGLE (program). Wikipedia. https://en.wikipedia.org/wiki/EAGLE_(program)

[7] Navabi, Z. (1993). VHDL: Analysis and modeling of digital systems (Vol. 2). New York: McGraw-Hill.