



Informe Final Dispositivo Dispensador de Gel

Maria Alejandra Morales Rincon, mamorales@unal.edu.co, Grupo 5
 Diego Fernando Gutiérrez García, dfgutierrezga@unal.edu.co, Grupo 6
 Cristian Eduardo Maldonado Samudio, cmaldonados@unal.edu.co, Grupo 6
 Electrónica Digital I
 Universidad Nacional de Colombia
 Bogotá
 15/12/2020

I. INTRODUCCIÓN

Tomando como punto de partida el avance del proyecto anteriormente presentado en el cual los conocimientos o pilares fundamentales para el desarrollo del proyecto se localizaban en el principio de lógica secuencial. Se tomó esta decisión ya que los problemas cotidianos que pueden ser tratados mediante la electrónica digital requieren esta área. Luego de determinar este punto de inicio se seleccionó una problemática que atañe a toda la población durante estos períodos recientes el cual es el aislamiento preventivo, la preservación de cada persona así como la reactivación tanto económica como de las rutinas cotidianas. Por esa razón se propone implementar un dispositivo dispensador de gel antibacterial o desinfectante cuya principal implementación ocurra en zonas universitarias o de ambiente académico. En este caso en concreto la Universidad Nacional. Sin duda esta iniciativa involucra una serie de actores, normas, leyes, mecanismos de solución, etc. Todo ello será abarcado en este documento.

II. PROBLEMÁTICA A RESOLVER

-Las personas que están físicamente cerca (dentro de los 6 pies de distancia) de una persona con COVID-19 o que tienen contacto directo con esa persona tienen mayor riesgo de infección.

-Las gotitas respiratorias también pueden depositarse sobre superficies y objetos. Es posible que una persona contraiga el COVID-19 al tocar una superficie u objeto que tenga el virus y luego tocarse la boca, la nariz o los ojos

III. ALTERNATIVAS DE SOLUCIÓN

De acuerdo a la organización mundial de la salud, el contagio por coronavirus se puede evitar de la siguiente manera:

- Lavar las manos con frecuencia.
- Usar agua y jabón o un desinfectante de manos a base de alcohol

-Mantener una distancia de seguridad con personas que tosen o estornudan.

-Utilizar mascarilla cuando no sea posible mantener el distanciamiento físico.

-No tocar los ojos, la nariz ni la boca.

-Cuando tosa o estornude, cubra la nariz y la boca con el codo flexionado o con un pañuelo.

-Si no se encuentra bien, es necesario quedarse en casa.

-En caso de tener fiebre, tos o dificultad para respirar, busque atención médica.

IV. OBJETIVOS

IV-A. Objetivo General

-Desarrollar un dispensador automático de gel antibacterial que cuente con medidor de temperatura y permita identificar a los estudiantes que han utilizado el dispositivo.

IV-B. Objetivo Específico

-Implementar el protocolo Uart en el funcionamiento del dispositivo dispensador.

-Implementar una pantalla LCD que permita informar a los usuarios las instrucciones de uso del dispositivo en tiempo real.

-Diseñar el contenedor del dispensador por medio de programas que permitan la impresión 3D.

V. MARCO TEÓRICO

V-A. Maquina de estados

Estar en un "estado" es la condición de una cosa en un tiempo determinado, algo que pueda realizar tareas y que utiliza estados como núcleo se les conoce como máquinas de estado. La clave para una máquina de estados es el concepto de tiempo e historia, ya que el estado de una máquina es evaluado periódicamente y cada vez que es evaluada, un nuevo estado

es elegido (también puede ser el mismo estado nuevamente) y el resultado es presentado.

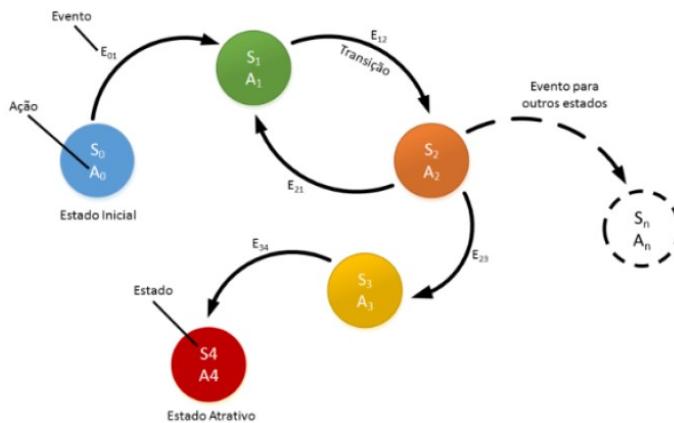


Figura 1: Ejemplo de una maquina de estados

V-B. Bomba de Aire

la bomba de aire es un motor conectado a un sistema de empuje mecánico la cual en promedio funciona con 9 - 12 V de entrada, por lo tanto para que funcione con la FPGA necesitamos una fuente de energía externa. Para esa conexión utilizamos un modulo puente H que es el que se va a encargar de hacer la conexión de la fuente de alimentación externa que en este caso fueron 14 V para poder regular la potencia y la corriente en los transistores internos. En el vídeo se muestra que el puente H también esta conectado a un arduino, este arduino simula la FPGA la cual nos entrega 3.3 V de salida en los pines, este solo le entrega estos 3.3 V al sistema de control del puente H. ya teniendo esta conexión al motor le llegan mas o menos 11 - 12 V con una potencia de 5 - 6 W haciéndolo funcionar.



Figura 2: Ejemplo de una bomba de aire de 12 V DC

V-C. LCD

Liquid crystal display o pantalla de cristal líquido es una pestaña delgada y plana formada por píxeles en color o monocromos colocados delante de una fuente de luz o reflectora. En su funcionamiento la LCD utiliza bajos valores de tensión. Existen LCDs con diferentes valores de resolución expresadas

en pixeles. Existen las pantallas HD que tienen una resolución nativa desde 1280x720 píxeles hasta 3840x2160 píxeles (4K UHD).

Cada pixel de una LCD es una capa de moléculas alineadas entre dos electrodos transparentes y dos electrodos de polarización cuya superficie tiene la función de ajustar las moléculas de cristal líquido en una dirección en particular aún antes de que el vampo electrico este presente , además tiene dos filtros de polarización, los ejes de transmisión que estan perpendiculares entre si. El cristal líquido evita que la luz que pasa por el primer filtro sea bloqueada por el segundo (cruzando) polarizador.

Cuando se aplica un voltaje a través de los electrodos, una fuerza de giro orienta las moléculas de cristal líquido paralelas al campo eléctrico, que distorsiona la estructura helicoidal



Figura 3: LCD

V-D. Proceso del dispensador

A continuación se presenta un esquema mediante un diagrama de estado el proceso que lleva a cabo el dispensador de gel teniendo las 3 entradas implementadas:

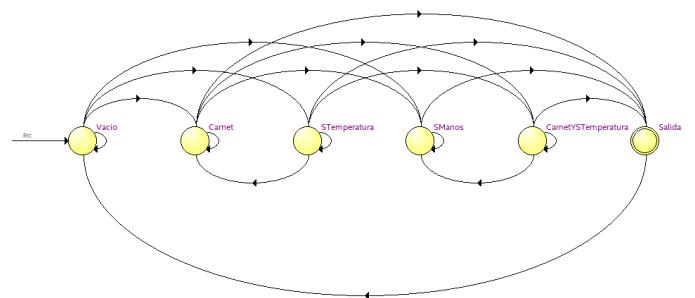


Figura 4: Diagrama de Estado código del proceso del dispensador


```
(11)
ACERCAR MANO
41 43 45 52 43 41 52 20 4d 41 4e 4f

PROCESANDO...
50 52 4f 43 45 53 41 4e 44 4f 2e 2e 2e

EL RESULTADO ES:
45 4c 20 52 45 53 55 4c 54 41 44 4f 20 45 53 3a

(12)
TEMPERATURA ELEVADA
54 45 4d 50 45 52 41 54 55 52 41 20 45 4c 45 56 41 44 41

ALERTA
41 4c 45 52 54 41

ESPERE...
45 53 50 45 52 45 2e 2e 2e

(13)
TEMPERATURA ACEPTABLE
54 45 4d 50 45 52 41 54 55 52 41 20 41 43 45 50 54 41 42 4c 45

PROCESO EXITOSO
50 52 4f 43 45 53 4f 20 45 58 49 54 4f 53 4f

ESPERE...
45 53 50 45 52 45 2e 2e 2e
```

Figura 11: Procesos 11, 12 y 13 de la LCD

```
(14)
TEMPERATURA INVALIDA
54 45 4d 50 45 52 41 54 55 52 41 20 49 4e 56 41 4c 49 44 41

PROCESO INVALIDO
50 52 4f 43 45 53 4f 20 49 4e 56 41 4c 49 44 4f

ESPERE...
45 53 50 45 52 45 2e 2e 2e

(15)
PROCESO TERMINADO
50 52 4f 43 45 53 4f 20 54 45 52 4d 49 4e 41 44 4f

GRACIAS!
47 52 41 43 49 41 53 21

VUELVA PRONTO!
56 55 45 4c 56 41 20 50 52 4f 4e 54 4f 21
```

Figura 12: Procesos 14 y 15 de la LCD

```
(16)
OPCION INVALIDA
4f 50 43 49 4f 4e 20 49 4e 56 41 4c 49 44 41

REINICIE PROCESO
52 45 49 4e 49 43 49 45 20 50 52 4f 43 45 53 4f

ESPERE...
45 53 50 45 52 45 2e 2e 2e
```

Figura 13: Último proceso de la LCD

A Continuación se prosigue a seguir con la declaración de variables y procesos para el display y la bomba:

```
55 begin
  |LCD_EN <= C1k_out ;
  |LCD_RW <= '0';
  |process(CLK);
  |variable n1:integer range 0 to 19999;
  |begin
    |if rising_edge(CLK) then
    |  n1:=n1+1;
  |else
    |  n1:=0;
  |end if;
  |end if;
  |end process;
  |LCD_Clk <= C1k_out;
  |process(CLk_out);
  |variable n2:integer range 0 to 499;
  |begin
    |if rising_edge(CLk_out) then
    |  n2:=n2+1;
  |else
    |  n2:=0;
  |end if;
  |end if;
  |end process;
  |process(CLk1);
  |variable n3:integer range 0 to 14;
  |begin
    |if rising_edge(CLk1) then
    |  n3:=n3+1;
  |else
    |  n3:=0;
  |end if;
  |end if;
  |end process;
  |process(CLk);
  |variable cmt1: std_logic_vector(4 downto 0);
  |begin
    |if Reset='0' then
    |  Current_State<=set_d1nf;
    |  cmt1<="11110";
    |  LCD_RS="0";
    |  if rising_edge(Lcd_CLK)then
    |    Current_State <> Current_State ;
    |  end if;
  |end if;
  |end if;
  |end process;
  |process(CLk);
  |variable cmt1: std_logic_vector(4 downto 0);
  |begin
    |if Reset='0' then
    |  Current_State<=set_d1nf;
    |  cmt1<="11110";
    |  LCD_RS="0";
    |  if rising_edge(Lcd_CLK)then
    |    Current_State <> Current_State ;
    |  end if;
  |end if;
  |end if;
  |end process;
```

Figura 14: Continuación declaración variables y procesos Display

```
109 case Current_State is
110 when set_d1nf=>
111   cnt1:="00000";
112   LCD_Data<="00000001"; -- /*?????????*/
113   Current_State<=set_cursor;
114 when set_cursor=>
115   LCD_Data<="00111000"; --/*?????8????,2??,5??*/
116   Current_State<=set_dcb;
117 when set_dcb=>
118   LCD_Data<="00001100"; --/*?????????,?,????,?????*/
119   Current_State<=set_cgram;
120 when set_cgram=>
121   LCD_Data<="00000110";
122   Current_State<=write_cgram;
123   when write_cgram=>
124     LCD_RS<'0';
125   if m="00" then
126     |LCD_Data<=cgram1(conv_integer(cnt1));
127   elsif m="01" then
128     |LCD_Data<=cgram2(conv_integer(cnt1));
129   elsif m="10" then
130     |LCD_Data<=cgram3(conv_integer(cnt1));
131   else
132     |LCD_Data<=cgram4(conv_integer(cnt1));
133   end if;
134   Current_State<=set_ddram;
135   when set_ddram=>
136     |cnt1<"11110" then
137       |cnt1:=cnt1+1;
138     |cnt1:="00000";
139   end if;
140   if cnt1<"01111" then
141     |LCD_Data<="1000000"+cnt1--80H
142   else
143     |LCD_Data<="11000000"+cnt1="10000";--80H
144   end if;
145   Current_State<=write_LCD_Data;
146   when write_LCD_Data=>
147     LCD_Data<="00000000";
148     Current_State<=set_cursor;
149     when others => null;
150   end case;
151 end if;
152 end process;
```

Figura 15: Inicio proceso de conversión de datos al Display

```

157
158     salida(0) <= brightness1;
159     reset1 <= not Entrada1(0);
160
161     --Divisor de frecuencia de 1 KHz
162     process(CLK, clk_1, contadorA)
163     begin
164         if CLK'event and CLK = '1' then
165             if contadorA < 50000 then
166                 contadorA <= contadorA +1;
167             else
168                 contadorA <= 0;
169                 clk_1 <= not(clk_1);
170             end if;
171         end if;
172     end process;
173
174     --clk_out <= clk_1;
175
176     ---Bomba
177     process(clk_1, reset1)
178     begin
179         if rising_edge(clk_1) then
180             count <= count +1;
181             if count(10) = '1' then --ciclo de trabajo de 30%
182                 brightness1 <= '1';
183             else
184                 brightness1 <= '0';
185             end if;
186         end if; -- end of sync
187
188         if reset1 = '1' then
189             count <= (others => '0');
190             brightness1 <= '0';
191         end if;
192     end process;
193     clk_out1 <= clk_1;
194
195     process(key_in)
196     begin
197         led_out <= (others => '1'); --??????????
198
199     case key_in is
200     when "1" => led_out <= "1"; -- key1????KEY1????LED?
201
202     when others => NULL;
203     end case;
204     end process;
205
206 end Behavioral;

```

Figura 16: Código de proceso de la bomba

```

16  entity ProyectoGel is
17      Port ( X : in STD_LOGIC;
18             Rst : in STD_LOGIC;
19             Clk : in STD_LOGIC;
20             Sal : out STD_LOGIC );
21  end ProyectoGel;
22
23  architecture Maquina of ProyectoGel is
24      type ESTADOS is (Vacio,STemperatura, Carnet, CarnetySTemperatura, SManos, Salida);
25      attribute syn_encoding: string;
26      attribute syn_encoding of ESTADOS: type is "sequential";
27      signal EstadosSig: ESTADOS;
28  begin
29
30     process(Rst,Clk) --Inicia el proceso secuencial
31     begin
32         if (Rst='1') then
33             Estados <- Vacio;
34         elsif (CLK'event and CLK='1') then
35             Estados <- EstadosSig;
36         end if;
37     end process;
38
39     process (Estados,X,Y)
40     begin
41         EstadosSig <- Estados;
42
43         case Estados is
44             when Vacio => if X='0' and Y='1' then EstadosSig<=STemperatura;
45             elsif X='1' and Y='0' then EstadosSig<=Carnet;
46             elsif X='1' and Y='1' then EstadosSig<=SManos;
47             end if;
48
49             when STemperatura => if X='0' and Y='1' then EstadosSig<=Carnet;
50             elsif X='1' and Y='0' then EstadosSig<=STemperatura;
51             elsif X='1' and Y='1' then EstadosSig<=Salida;
52             end if;
53
54             when Carnet => if X='0' and Y='1' then EstadosSig<=CarnetySTemperatura;
55             elsif X='1' and Y='0' then EstadosSig<=SManos;
56             elsif X='1' and Y='1' then EstadosSig<=Salida;
57             end if;
58
59             when CarnetySTemperatura => if X='0' and Y='1' then EstadosSig<=SManos;
60             elsif X='1' and Y='0' then EstadosSig<=Carnet;
61             elsif X='1' and Y='1' then EstadosSig<=Salida;
62         end if;
63
64         when SManos => if X='1' or Y='1' then EstadosSig<=Salida;
65         when Salida => EstadosSig<=Vacio;
66     end process;
67
68     Sal <= '1' when (Estados = Salida) else '0';
69 end Maquina;

```

Figura 17: Proceso máquina de estados

Cabe resaltar que se asignaron unos pines para que la LCD 16X2 y la bomba de aire trabajaran con la FPGA, los pines fueron los siguientes:

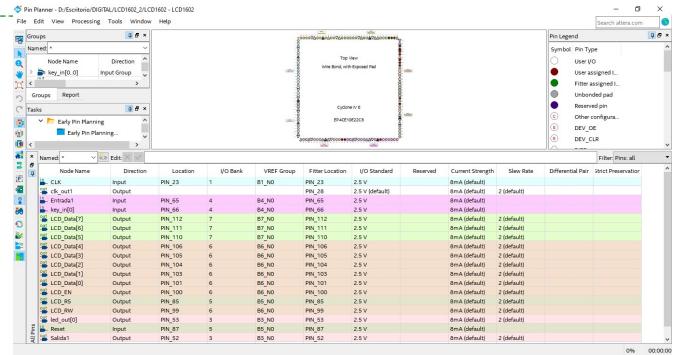


Figura 18: PIN PLANNER

VI-A. Código de Arduino

```

#include <SoftwareSerial.h> //libreria de comunicacion serial uart
#include <mlx90615.h> // libreria para el sensor de temperatura a distancia
#include <Wire.h> // averiguar
#include <SPI.h> // averiguar
#include <MFRC522.h>

const int Trigger = 2; //Pin digital 2 para el Trigger del sensor
const int Echo = 3; //Pin digital 3 para el Echo del sensor

#define RST_PIN 49 //Pin 49 para el reset del RC522
#define SS_PIN 53 //Pin 53 para el SS (SDA) del RC522

MFRC522 mfrc522(SS_PIN, RST_PIN); //Creamos el objeto para el RC522
MLX90615 mlx = MLX90615(); //revisar 20 y 21 definimos el sensor de la libreria

```

MLX90615 mlx = MLX90615(); //revisar 20 y 21 definimos el sensor de la libreria

Figura 19: Definición de librerías y pines.

```

SoftwareSerial TEMPERATURA(19,18); // TX esto lo usamos para definir una nueva linea de datos que sera la que transmira la informacion del sensor de temperatura
SoftwareSerial RFID(17,16); // RX esto lo usamos para definir una nueva linea de datos que sera la que transmira la informacion del rfid
SoftwareSerial ULTRASONICO(D10,19); // TX esto lo usamos para definir una nueva linea de datos que sera la que transmira la informacion del sensor de ultro sonido
int lectura;
void setup () {
    TEMPERATURA.begin (9600); //comunicacion serial define la velocidad de comunicacion del sensor de temperatura
    RFID.begin (9600); //comunicacion serial define la velocidad de comunicacion del rfid
    ULTRASONICO.begin (9600); //comunicacion serial define la velocidad de comunicacion del sensor de ultrasonido
    Serial.begin (9600); //comunicacion serial para el computador
}

```

Figura 20: Definición de pines de communication serial para los perifericos.

```

mlx.begin(); // como el sensor es i2c pues debemos leerlo
SPI.begin(); // Iniciamos el Bus SPI
mfrc522.PCD_Init(); // Iniciamos el MFRC522
Serial.println("Control de objeto:");
pinMode(Trigger, OUTPUT); //pin como salida
pinMode(Echo, INPUT); //pin como entrada
digitalWrite(Trigger, LOW); //Inicializamos el pin con 0
pinMode(22, OUTPUT);
pinMode(24, OUTPUT);
pinMode(26, OUTPUT);

byte ActualUID[4]; //almacenara el codigo del Tag leido
byte Usuario1[4] = {0x08, 0x8C, 0xF, 0x51}; //CAMBIAR EL CODIGO USEN EL MONTAJE BASE Y LEEN LAS TARJETAS
byte Usuario2[4] = {0xB5, 0x96, 0x6F, 0x51}; //CAMBIAR EL CODIGO USEN EL MONTAJE BASE Y LEEN LAS TARJETAS
byte Usuario3[4] = {0xD7, 0xDD, 0xA8, 0xE4}; //CAMBIAR EL CODIGO USEN EL MONTAJE BASE Y LEEN LAS TARJETAS
void loop () {
    //entregara en el computador la lectura del sensor de temperatura
    // Serial.print("Ambient = ");
    // Serial.print(mlx.get_ambient_temp());
    // Serial.print(" *CtoObject = ");
    int x = mlx.get_object_temp();
    // Serial.println(x, HEX); //nota mirar los decimales asci a hexadecimal
    // Serial.print("TEMPERATORA: ");
    // Serial.print(x);
    // Serial.println(" *C");
}

```

Figura 21: Lectura del modulo I2C y escritura sobre la pantalla serie de Arduino.

```

// Serial.println(" *C");
// finaliza lo del pc pasamos a trasmisirlo en la linea 1
//TEMPERATURA.print(x); //ESTO HACE QUE SE ESCRIBA POR UART A TRAVES DE LOS PINES LA INFORMACION
//TEMPERATURA.println(mix.get_object_temp(),BIN); ESTO ESCRIBIRA EN BINARIO LA SALIDA 0 Y 1
//INICIO DE ULTRASONIDO
long t; //tiempo que demora en llegar el eco
int d; //distancia en centimetros
digitalWrite(Trigger, HIGH);
delayMicroseconds(10);
digitalWrite(Trigger, LOW);
t = pulseIn(Echo, HIGH); //obtenemos el ancho del pulso
d = t/59; //escalamos el tiempo a una distancia en cm
//Serial.println(d);//esto lo enviara en codigo ASCII
//Serial.println(x);//esto lo enviara en codigo ASCII
delay(100);
//Hacemos una pausa de 100ms
//FIN
//RFID INICIO
if ( mfrc522.PICC_IsNewCardPresent() ) {
    //Seleccionamos una tarjeta
    if ( mfrc522.PICC_ReadCardSerial() ) {
        // Enviamos seriamente su UID
        // Serial.print(F("Card UID:"));
        for (byte i = 0; i < mfrc522.uid.size(); i++) {
            Serial.print(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " ");
            Serial.print(mfrc522.uid.uidByte[i], HEX); // esta
        }
        ActualUID[i]=mfrc522.uid.uidByte[i];
    }
}

```

Figura 22: Código implementado para el Ultrasonido.

```

Serial.print("      ");
//comparamos los UID para determinar si es uno de nuestros usuarios
if(compareArray(ActualUID,Usuario1)){
    // Serial.print("Acceso concedido...      ");
    // RFID.print("Acceso concedido");
    loc = 0;
}

//RFID.println(0,BIN); ESTO ESCRIBIRA EN BINARIO LA SALIDA 0 Y 1
}

if(compareArray(ActualUID,Usuario2)){
    // Serial.print("Acceso concedido...      ");
    // RFID.print("CONCEDIDO");
    //RFID.println(0,BIN); ESTO ESCRIBIRA EN BINARIO LA SALIDA 0 Y 1
    loc = 0;
}

else if(compareArray(ActualUID,Usuario3)){
    // Serial.print("Acceso concedido...      ");
    // RFID.print("CONCEDIDO");
    //RFID.println(0,BIN); ESTO ESCRIBIRA EN BINARIO LA SALIDA 0 Y 1
    loc = 0;
}

else{
    // Serial.print("Acceso denegado...      ");
    // RFID.print("Acceso denegado..."); 
}

```

Figura 23: Código implementado para la RFID

```

loc = 1;
//ACTIVAR LA SIGUIENTE LINEA ES MAS SENCILLA
//RFID.println(1,BIN); ESTO ESCRIBIRA EN BINARIO LA SALIDA 0 Y 1
// Terminamos la lectura de la tarjeta tarjeta actual
mfrc522.PICC_HaltA();
}

if((2<d<10)&&(x<39)&&(loc != 0)){
    ULTRASONIDO.print(d); //ULTRASONIDO.print(d,HEX);
    TEMPERATURA.print(x); //TEMPERATURA.print(x, HEX);
    RFID.print("SIGA");//RFID.print("SIGA");
}

if((d<15)&&(d>3)&&(x<38)&&(loc == 0)){//ULTIMO IF
    Serial.print("distancia: "); //ULTRASONIDO.print(d,HEX);
    Serial.print(d); //ULTRASONIDO.print(d,HEX), BIN
    ULTRASONIDO.print(d,HEX);
    Serial.print(" temperatura: "); //ULTRASONIDO.print(d,HEX);
    Serial.print(x+3); //TEMPERATURA.print(x, HEX);
    TEMPERATURA.print(x+3, HEX);
    Serial.print("      "); //ULTRASONIDO.print(d,HEX);
    Serial.println("acceso concedido");//RFID.print("SIGA");
    RFID.print(1, HEX);
}

```

Figura 24: Continuación del código de RFID.

```

// digitalWrite(22, HIGH); //CONDICIONAL
digitalWrite(24, HIGH); //CONDICIONAL
digitalWrite(26, HIGH); //CONDICIONAL
delay(1000);
loc = 2;
}
if(loc == 1){
    Serial.println("acceso denegado no esta en base de datos");//RFID.print("SIGA");
    digitalWrite(22, LOW); //CONDICIONAL
    digitalWrite(24, LOW); //CONDICIONAL
    digitalWrite(26, LOW); //CONDICIONAL
    delay(1000);
    loc = 2;
}
}

```

Figura 25

```

//Función para comparar dos vectores
boolean compareArray(byte array1[],byte array2[])
{
    if(array1[0] != array2[0]) return(false);
    if(array1[1] != array2[1]) return(false);
    if(array1[2] != array2[2]) return(false);
    if(array1[3] != array2[3]) return(false);
    return(true);
}

```

Figura 26

VII. RESULTADOS

A continuacion se muestran algunas fotos de lo que fueron las conexiones y el manejo de la FPGA antes de la ultima entrega:



Figura 27: Prueba de la bomba

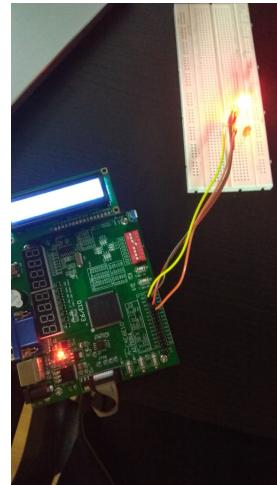


Figura 28: Foto del montaje.

Por último, se muestran fotos del resultado final del proyecto, el cual consta de 3 parte o "tapas".

La "tapa"de mas abajo, que es la que contiene el gel para no hacer daño a los componentes de mas arriba.

La "tapa"del medio, que es donde esta la FPGA y en nuestro caso un arduino MEGA, las conexiones y la LCD.

La "tapa"de arriba, que es en donde están los 3 sensores de entrada (RFID, Sensor de temperatura y sensor de proximidad). Ademas de que el espacio para meter la mano y hacer uso de estos dispositivos.

Cabe resaltar que hay un tubo que pasa al fondo que es el que conduce el gel antibacterial, el cual llega hasta la "tapa"de arriba, quedando justo encima del espacio para meter la mano.



Figura 29: Foto del montaje.



Figura 30: Foto del montaje.



Figura 31: Foto del montaje.



Figura 32: Foto del montaje.

VIII. CONCLUSIONES

El proceso de establecer un proyecto de aplicación de electrónica Digital con diverso elementos como la FPGA, una fuente de tensión o un Arduino puede ser extenuante. Esto debido a los objetivos o requerimientos que se deben cumplir con respecto a estándares de hardware y software. Es indispensable tratar con precaución cada herramienta para que no sufran averías en un periodo determinado. Por otra parte se reconoce la importancia de la comunicación de los diferentes lenguajes de programación que pueden verse involucrados, en este caso fueron el lenguaje UART, el VHDL y del Arduino que en la mayoría de ocasiones es C++. En el transcurso de la elaboración se reconoce como el ámbito que rodea al problema

que el dispositivo pretende resolver, siendo más concreto el sector de la salud, la situación sanitaria actual, sin omitir a los agentes y normas que pueden envolverse. Finalmente es importante ofrecer a la sociedad herramientas para resolver los problemas cotidianos presentados, más si son elementos que son primordiales en diversos sectores, sin embargo, el sector del campus universitario se plantea como un buen punto de inicio para reactivar el sector académico de manera presencial.

REFERENCIAS

- [1] T.L. Floyd. Fundamentos de Sistemas Digitales. Novena Edición. Madrid: Pearson Educación S.A, 2008.
- [2] D.M. Harris S.L.Harris. Digital Design and Computer Architecture
- [3] R.j. Tocci. Sistemas Digitales- Principios y aplicaciones. Décima Edición. México: Pearson Educación S.A, 2007.