Digital Laboratory II - Project Documentation

Prof. Dr. Paulo Cugnasca

Bancada B5

Otávio Felipe de Freitas - 11261249

William Abe Fukushima - 11261771

Gabriel Pereira de Carvalho - 11257668

December 6, 2021

Contents

Part I

Week 1: Theme Definition and Requirements

Chapter 1

Theme

1.1 Theme Proposal

The project envisioned by the group is a security system with proximity alert and motion sensor based on the `Sonar` circuit developed in previous experiments.

The system to be developed will be positioned **inside the residence** with a view of the door or entry point to be monitored.

The proposal is to provide some security options depending on the user's location – *at home* or *away from home* –, which can be selected through interaction with `MQTT`.

1.1.1 At Home Mode

When in *at home* mode, the `HC-SR04` component can be used to detect movement at the door.

It will be possible to signal the resident through `MQTT`, useful when the resident is away from the terminal, and through LEDs, providing easy and noticeable signaling if someone is near the terminal.

After being notified, the resident at home could then take appropriate action. For example, if the resident was expecting a visitor, they could go to the door and greet the visitor. In another example, if the resident was not expecting a visitor, they could call the police upon receiving the system's notification.

1.1.2 Away from Home Mode

In *away from home* mode, we assume that any approach or movement is unwanted.

To alert the neighborhood, a buzzer can be activated, making enough noise to prompt action. Thus, we would be integrating a new component into the base project.

There needs to be a signaling for the resident who is absent through communication with their mobile device.

Figure 1.1: System offers easily configurable security options for the user

1.1.3 Additional Features

It is also intended to use passwords to arm and disarm the security system, as well as to terminate the signaling of any functionality if it has been triggered accidentally.

So that the resident can monitor the operating history of the system, we intend to write logs corresponding to each sensor trigger to a RAM.

Chapter 2

Sprint Week 2

2.1 Basic Journey

Requirements gathering and basic implementation to be carried out during the first week of the project.

2.1.1 Functional Requirements

The proposed system in security mode must identify if there has been any movement inside a room and enter an alert mode if detection occurs, triggering a transmission of the captured data and time when the movement was detected to the client's cell phone and a buzzer alarm.

2.1.2 Non-functional Requirements

To detect movement, the system must also consider the measurement inaccuracies and the environment condition to generate sensor sensitivity. If the environment is closed and static, a higher sensitivity can be used, but if it is outdoors, for example, the sensitivity must be adjusted so that the system does not enter alert mode with small movements.

2.1.3 Physical Requirements

Initially, components from the Kit Home Lab are required to interact with the physical environment of the Digital Laboratory II discipline. As the sonar will be reused for the group's project, the SG90 servo motor and HC-SR04 sensor are required, in addition to the Wemos D1 R1 board, which has the ESP8266 component. To mediate communication with the laboratory's FPGA, the `MQTT Dash` application will be used.

2.1.4 Implementation

The first modification is to transform the circuit to enable motion detection in a room. Therefore, using the sonar circuit, a memory can be added

to store the distances measured in a first sweep, thus detecting movement if any measurement deviates from a certain range relative to the first measurement for that point in question.

Algorithm 1: Motion Detection

```
while System is on do
  if Circuit was off then
    Reset servo motor position
    while First Sweep do
      HC-SR04 measures distance
      Store measured distance in memory
      Change servo motor position
    end while
  else
    HC-SR04 measures distance
    Compare with the stored measurement for that position plus system sensitivity
    if Measurement outside the range then
      Detects motion
      Enters alert state
    else
      Change servo motor position
    end if
  end if
end while
```

Another necessary modification is the transmission of the detection time. For this, an integration will be made with a Python script that passes the updated time to the UART.

Chapter 3

Sprint Week 3

3.1 Service Monitoring

There are three functionalities to be implemented in the third week: "At Home" mode, "Away from Home" mode, and presence detection at the entrance.

3.1.1 At Home

When arming the security system with the "At Home" mode, it is desired that the motion detector, developed in the previous week, scans the room in which it is installed and interacts with the `MQTT Dash` to signal to the resident if there is any movement. In this case, it is not necessary for the signaling to be more incisive, so the change of state of a widget is sufficient.

3.1.2 Away from Home

When the resident is absent, the system should scan the room and, if it detects any movement according to the established sensitivity, notify through the `MQTT Dash` and activate a buzzer to alert the neighborhood.

When the security system is alarmed, the circuit must store the time at which the trigger occurred. This functionality will be integrated with the *Jupyter* through `MQTT`.

3.1.3 Door Presence Alert

This function works in the opposite way, that is, the sensor that monitors the entrance of the house should notify that there is something present. Motion detection is used, and if there is a detection for a determined period of time, notify the resident through the `MQTT`.

3.1.4 Implementation

It is intended to implement the functions described above according to the pseudocodes 2 and 3.

Algorithm 2: Description of Modes

```
while turn on do
    Activate motion detection circuit
    if chosen mode is "At Home"
    and Circuit detects motion then
        Notify the resident via MQTT
    else if chosen mode is "Away
    from Home" and Circuit detects
    motion then
        Notify the resident via MQTT
        Activate the Buzzer
        Store time
    end if
end while
```

Algorithm 3: Exterior Detection

```
while turn on do
    Activate motion detection circuit
    if Circuit detects motion for a
    determined time then
        Notify the resident via MQTT
    end if
end while
```

Chapter 4

Sprint Week 4

4.1 Password Protection

In the last week available for the project, the security system will be integrated with a password.

To simplify the hardware, it is desired to take advantage of the IoT environment and perform integration with a password using *Jupyter*. Thus, it is not necessary to have memory in the hardware to store the password and also prevents it from being lost in case of power failure.

The security system should require a password to activate and deactivate, as well as, in case of motion detection in the "Away from Home" mode, to turn off the buzzer, the resident must also provide the password. This, which would be checked on the server in a real scenario, will be sent through the serial transmission of the circuit, and the Python code will be responsible for responding if the authentication is correct.

Part II

Week 2: Basic Journey

Chapter 5

Objectives

The objective of the first sprint of the project was to implement the basic user journey, trying to meet the basic functional requirements of the system.

Chapter 6

Project Documentation

6.1 Project Description

The proposal of this first sprint is to transform the circuit to enable motion detection in a room. Therefore, using the sonar circuit, a memory can be added to store the distances measured in a first scan, thus detecting motion if any measurement deviates from a certain interval in relation to the first measurement for that point.

Algorithm 4: Detecção de Movimento

```
while Sistema ligado do
  if Circuito estava desligado then
    Reseta posição do servo motor
    while Primeiro Varrimento do
      HC-SR04 mede distância
      Armazena distância medida na memória
      Muda posição do servo motor
    end while
  else
    HC-SR04 mede distância
    Compara com a medida armazenada para aquela posição somada à
    sensibilidade do sistema
    if Medida fora do intervalo then
      Detecta movimento
      Entra em estado de alerta
    else
      Muda posição do servo motor
    end if
  end if
end while
```

6.2 Control Unit

The state machine [15.1] has undergone changes compared to the sonar circuit. The new state machine includes system calibration and motion detection. During the calibration state, the circuit triggers writing to memory, and the distance stored will be compared with the measurement.

If the measured distance is less than the stored distance by a specified interval, there will be a motion alert. Currently, we are using an interval in the form of $[dist; dist + sensitivity]$. In the future, we intend to offer sensitivity as a configurable option for the user.

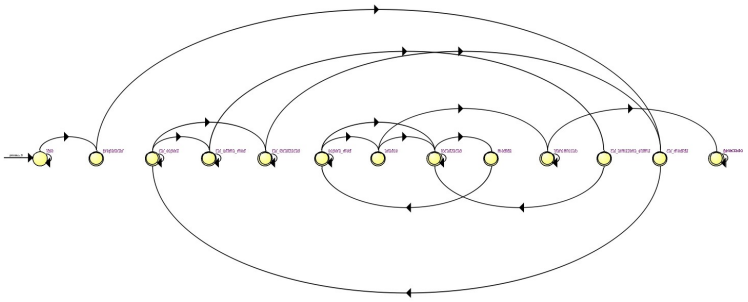

Figure 6.1: Modified state machine

6.3 Data Flow

Few components were added in the data flow to achieve the motion detection goal. To store the distances measured during calibration, a single-port RAM and a counter to indicate the end of the scan were implemented – it can be used to indicate larger numbers of scans and thus perform an average. The signal indicating the last position of the sensor – *updown counter* – is the input of an edge detector, as it indicates the count of scans.

In the data flow of figure [6.3], we also observe adders used to calculate the sensitivity interval where the alert mode is not selected.

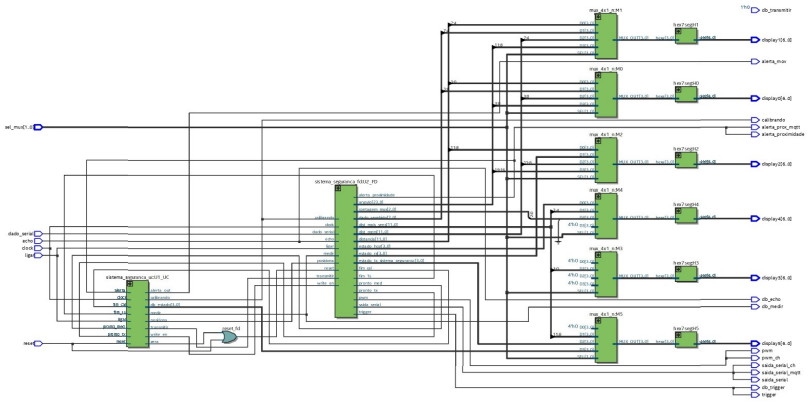

Figure 6.2: Complete circuit Data Flow. There were no changes in higher-level components, changes were made within the Control Unit [6.2] and Data Flow.

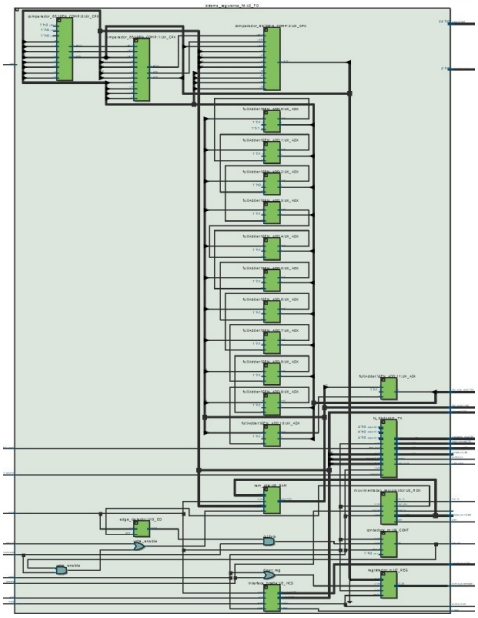

Figure 6.3: RTL Diagram of the Data Flow. There is an edge detector in the bottom left part, which receives as input the last position signaling from the *updown counter*. The output of the edge detector is part of the logic to write to the RAM memory that stores the distances measured during circuit calibration.

6.4 Pinout

In order to perform tests with the LabEAD infrastructure, the project pinout was made as presented in this section.

Sinal	Direção	Localização	Referência
alerta_mov	Output	PIN_H16	MQTT S0
alerta_proximidade	Output	PIN_AA2	LEDR0
calibrando	Output	PIN_L1	LEDR9
clock	Input	PIN_M9	
dado_serial	Input	PIN_R21	
db_echo	Output	PIN_N2	LEDR4
db_medir	Output	PIN_AA1	LEDR1
db_transmitir	Output	PIN_W2	LEDR2
db_trigger	Output	PIN_Y3	LEDR3
display0-5	Output		HEX0-5
echo	Input	PIN_T17	
ligar	Input	PIN_B16	E1
pwm	Output	PIN_K16	
pwm_ch	Output	PIN_F15	Scope Channel 1
reset	Input	PIN_N16	E0
saida_serial	Output	PIN_P18	Protocol
saida_serial_ch	Output	PIN_F12	Scope Channel 2
saida_serial_mqtt	Output	PIN_A13	
sel_mux[0]	Input	PIN_M16	E2
sel_mux[1]	Input	PIN_C16	E3
trigger	Output	PIN_J17	

6.5 Modelsim Simulation

The code up to the time of delivery of this planning was simulated in *Modelsim*. The desired functionality is not yet perfect. Storing distances in RAM during calibration works correctly, however, comparison and alert need to be improved, as there is a considerable difference between sensor measurements.

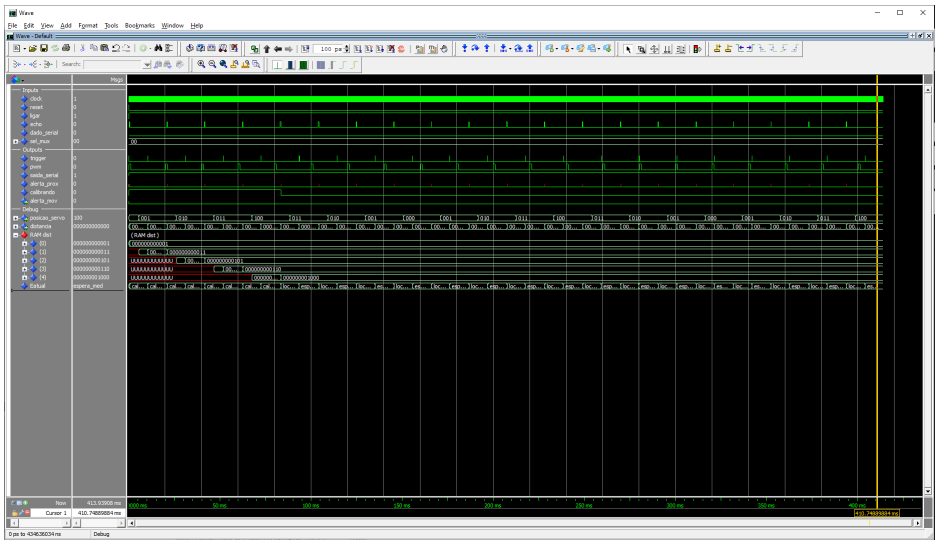

Figure 6.4: The simulation shows the measurements being saved in RAM during calibration. As it is a simulation, the measurements are perfect, something that does not happen in reality. Therefore, a way to consider reality must be found during code implementation. Two alternatives are considered: trigger motion alert if less than 90% of measurements are incorrect or reduce the sensor sweep angle and consider it in an optimal testing environment – an environment without corners and without attempts to measure distance at large angles.

Chapter 7

LabEAD Demonstration

During the class, tests were conducted using the LabEAD infrastructure to validate the week's sprint.

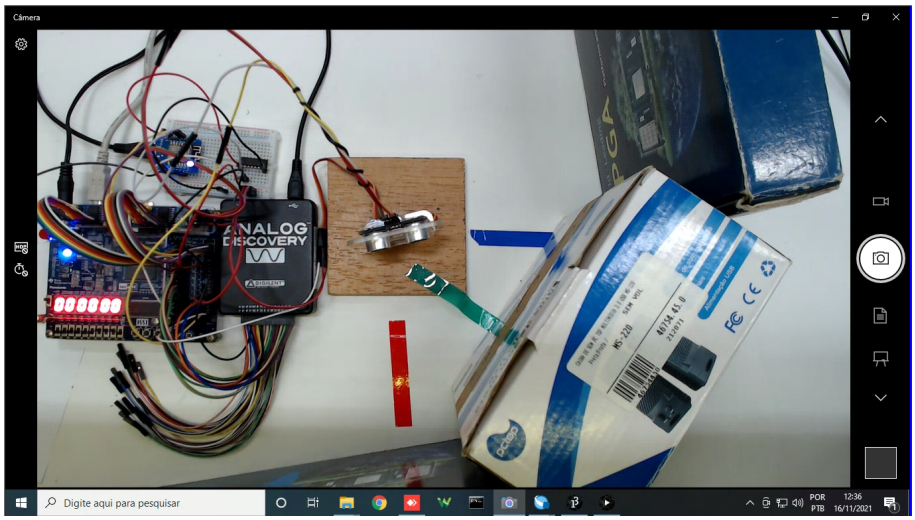

Figure 7.1: Initial test conditions

Initially, the reset end was triggered. As we saw in the state machine, when the reset is triggered, the circuit is placed in the `idle` state. The servo motor remains in its initial position, and no system functionality is activated.

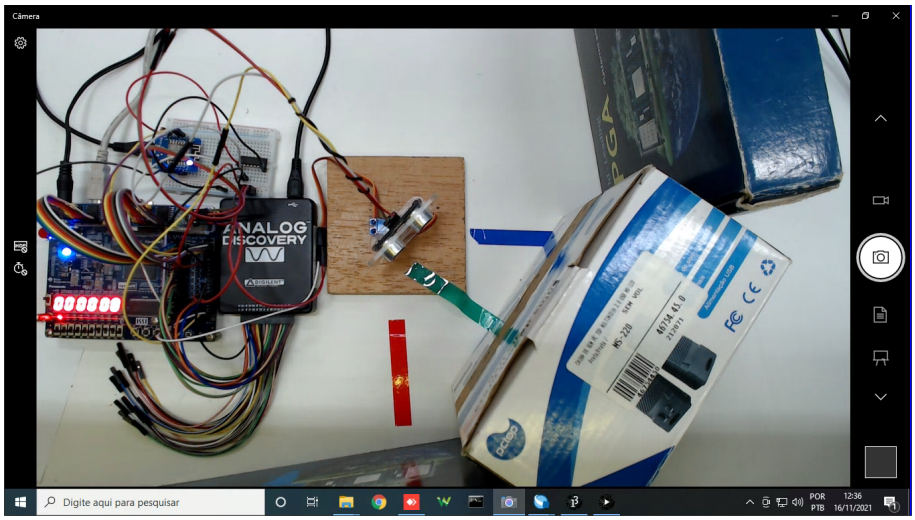

Figure 7.2: Calibration stage

After triggering the `ligar` input, the circuit performs an initial sweep collecting distance information for each of the 25 servo motor positions. In figure [7.2], we see the LEDR9 output activated indicating that the circuit is calibrating.

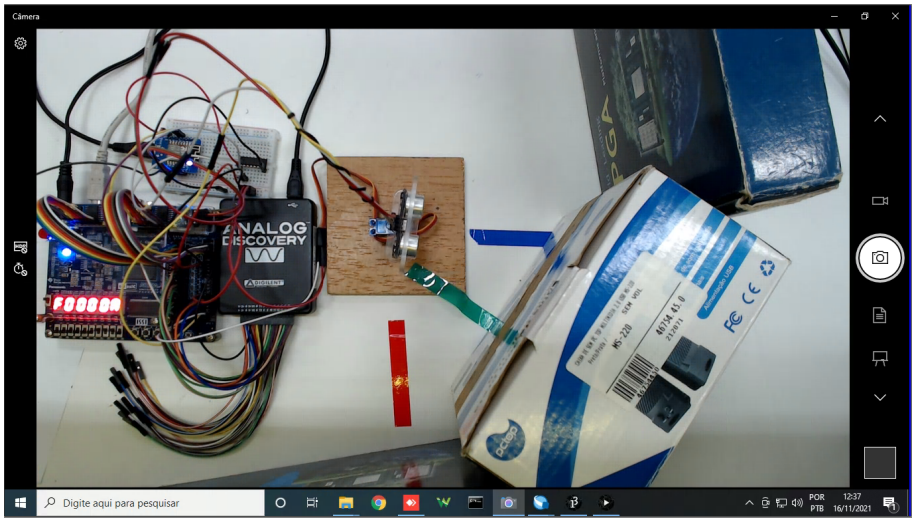

Figure 7.3: Traversing RAM memory

Using signal multiplexing, it is possible to display the output of the `ram_dist` block to check the content of each position in memory. In image [7.3], using `sel_mux` equal to 10, the memory content is displayed on displays `HEX0-2`.

It is worth noting that unlike what happened in the sonar project, distances are stored in binary representation and not BCD. Since we need to perform addition operations with the sensitivity value, we decided to make this change in the project.

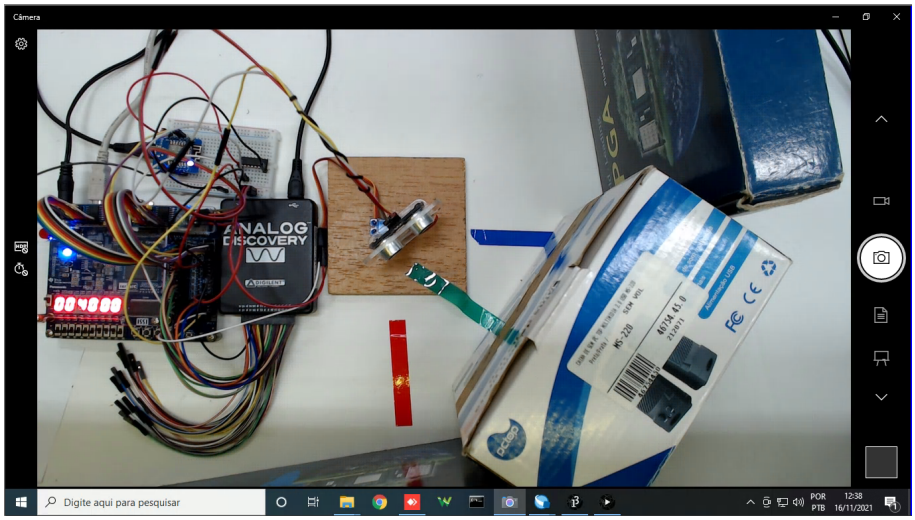

Figure 7.4: Motion detection

The `S0` output of the MQTT Dash was configured to be activated in case of motion detection. This basic functionality will be expanded in the next sprint of the project.

During the demonstrations, we found that the measurements provided by the sensor were not very stable. Even with a stationary bench setup, we observed small variations in measurements during sensor sweeps. Dealing properly with these inaccuracies will be a project priority in the future.

Chapter 8

Project Codes

8.1 Security System

8.1.1 Complete Circuit

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity sistema_seguranca is
  port(
    -- Inputs
    clock       : in std_logic;
    reset       : in std_logic;
    ligar       : in std_logic;
    echo        : in std_logic;
    dado_serial : in std_logic;
    sel_mux     : in std_logic_vector(1 downto 0);
    -- Outputs
    trigger            : out std_logic;
    db_trigger         : out std_logic;
    db_echo            : out std_logic;
    pwm                : out std_logic;
    saida_serial       : out std_logic;
    saida_serial_ch    : out std_logic;
    saida_serial_mqtt  : out std_logic;
    pwm_ch             : out std_logic;
    alerta_proximidade : out std_logic;
    alerta_prox_mqtt   : out std_logic;
    db_transmitir      : out std_logic;
    db_medir           : out std_logic;
    calibrando         : out std_logic;
    alerta_mov         : out std_logic;
    display0           : out std_logic_vector(6 downto 0);
    display1           : out std_logic_vector(6 downto 0);
    display2           : out std_logic_vector(6 downto 0);
    display3           : out std_logic_vector(6 downto 0);
    display4           : out std_logic_vector(6 downto 0);
```

```
34      display5           : out std_logic_vector(6 downto 0)
35    );
36  end entity;
37
38  architecture sistema_seguranca_arch of sistema_seguranca is
39
40    -- Unidade de Controle
41    component sistema_seguranca_uc is
42      port (
43        -- Inputs
44        clock      : in std_logic;
45        reset      : in std_logic;
46        ligar      : in std_logic;
47        fim_1s     : in std_logic;
48        pronto_med : in std_logic;
49        pronto_tx  : in std_logic;
50        fim_cal    : in std_logic;
51        alerta     : in std_logic;
52        -- Outputs
53        zera       : out std_logic;
54        posiciona  : out std_logic;
55        medir      : out std_logic;
56        transmitir : out std_logic;
57        calibrando : out std_logic;
58        write_en   : out std_logic;
59        alerta_out : out std_logic;
60        db_estado  : out std_logic_vector(3 downto 0)
61      );
62    end component;
63
64    -- Fluxo de Dados
65    component sistema_seguranca_fd is
66      port (
67        -- Inputs
68        clock       : in std_logic;
69        reset       : in std_logic;
70        ligar       : in std_logic;
71        medir       : in std_logic;
72        posiciona   : in std_logic;
73        transmitir  : in std_logic;
74        echo        : in std_logic;
75        dado_serial : in std_logic;
76        calibrando  : in std_logic;
77        write_en    : in std_logic;
78        -- Outputs
79        pwm                        : out std_logic;
80        trigger                    : out std_logic;
81        saida_serial               : out std_logic;
82        pronto_tx                  : out std_logic;
83        alerta_proximidade         : out std_logic;
84        fim_1s                     : out std_logic;
85        meio_1s                    : out std_logic;
86        pronto_med                 : out std_logic;
87        fim_cal                    : out std_logic;
88        contagem_mux               : out std_logic_vector(2 downto
      0);
89        estado_hcsr                : out std_logic_vector(3 downto
```

```
    0);
90      estado_tx_sistema_seguranca : out std_logic_vector(3 downto
    0);
91      estado_rx                   : out std_logic_vector(3 downto
    0);
92      estado_tx                   : out std_logic_vector(3 downto
    0);
93      posicao_servo               : out std_logic_vector(4 downto
    0);
94      dado_recebido               : out std_logic_vector(7 downto
    0);
95      distancia                   : out std_logic_vector(11 downto
    0);
96      dist_mem                    : out std_logic_vector(11 downto
    0);
97      dist_mais_sens              : out std_logic_vector(11 downto
    0);
98      angulo                      : out std_logic_vector(23 downto
    0)
99    );
100 end component;
101
102 -- Multiplexador 4x1
103 component mux_4x1_n is
104   generic (
105     constant BITS: integer := 4
106   );
107   port (
108     D0 :     in  std_logic_vector (BITS-1 downto 0);
109     D1 :     in  std_logic_vector (BITS-1 downto 0);
110     D2 :     in  std_logic_vector (BITS-1 downto 0);
111     D3 :     in  std_logic_vector (BITS-1 downto 0);
112     SEL:     in  std_logic_vector (1 downto 0);
113     MUX_OUT: out std_logic_vector (BITS-1 downto 0)
114   );
115 end component;
116
117 -- Display de 7 segmentos
118 component hex7seg is
119   port (
120     hexa : in  std_logic_vector(3 downto 0);
121     sseg : out std_logic_vector(6 downto 0)
122   );
123 end component;
124
125 -- Sinais
126 signal pronto_tx_s                              : std_logic;
127 signal fim_1s_s, zera_s, reset_fd               : std_logic;
128 signal meio_1s_s                    : std_logic;
129 signal posiciona_s, medir_s                     : std_logic;
130 signal saida_serial_s, pwm_s                    : std_logic;
131 signal alerta_proximidade_s                     : std_logic;
132 signal transmitir_s, pronto_med_s               : std_logic;
133 signal fim_cal_s, calibrando_s, write_en_s      : std_logic;
134 signal contagem_mux_3bits                      :
    std_logic_vector(2 downto 0);
135 signal contagem_mux_4bits                      :
```

```
    std_logic_vector(3 downto 0);
  signal sistema_seguranca_estado, posicao_4bits  :
    std_logic_vector(3 downto 0);
  signal estado_hcsr, estado_tx_sistema_seguranca :
    std_logic_vector(3 downto 0);
  signal estado_rx, estado_tx                     :
    std_logic_vector(3 downto 0);
  signal posicao_5bits                            :
    std_logic_vector(4 downto 0);
  signal dado_recebido_s                          :
    std_logic_vector(7 downto 0);
  signal distancia_bcd                            :
    std_logic_vector(11 downto 0);
  signal dist_mem                                 :
    std_logic_vector(11 downto 0);
  signal dist_sens                                :
    std_logic_vector(11 downto 0);
  signal angulo_bcd_hex                           :
    std_logic_vector(23 downto 0);
  signal trigger_s                  : std_logic;

  -- Saidas dos multiplexadores
  signal m0_out, m1_out, m2_out, m3_out, m4_out, m5_out :
    std_logic_vector(3 downto 0);

begin

  -- Logica de sinais
  reset_fd           <= reset or zera_s;
  contagem_mux_4bits <= '0' & contagem_mux_3bits;

  -- Instancias
  U1_UC: sistema_seguranca_uc
    port map(
      -- Inputs
      clock      => clock,
      reset      => reset,
      ligar      => ligar,
      fim_1s     => fim_1s_s,
      pronto_med => pronto_med_s,
      pronto_tx  => pronto_tx_s,
      fim_cal    => fim_cal_s,
      alerta     => alerta_proximidade_s,
      -- Outputs
      zera       => zera_s,
      posiciona  => posiciona_s,
      medir      => medir_s,
      transmitir => transmitir_s,
      calibrando => calibrando_s,
      write_en   => write_en_s,
      alerta_out => alerta_mov,
      db_estado  => sistema_seguranca_estado
    );

  U2_FD: sistema_seguranca_fd
    port map(
      -- Inputs
```

```
182       clock       => clock,
183       reset       => reset_fd,
184       ligar       => ligar,
185       medir       => medir_s,
186       posiciona   => posiciona_s,
187       transmitir  => transmitir_s,
188       echo        => echo,
189       dado_serial => dado_serial,
190       calibrando  => calibrando_s,
191       write_en    => write_en_s,
192       -- Outputs
193       pwm                          => pwm_s,
194       trigger                      => trigger_s,
195       saida_serial                 => saida_serial_s,
196       pronto_tx                    => pronto_tx_s,
197       alerta_proximidade           => alerta_proximidade_s,
198       fim_1s                       => fim_1s_s,
199       meio_1s                      => meio_1s_s,
200       pronto_med                   => pronto_med_s,
201       fim_cal                      => fim_cal_s,
202       contagem_mux                 => contagem_mux_3bits,
203       estado_hcsr                  => estado_hcsr,
204       estado_tx_sistema_seguranca => estado_tx_sistema_seguranca,
205       estado_rx                    => estado_rx,
206       estado_tx                    => estado_tx,
207       posicao_servo                => posicao_5bits,
208       dado_recebido                => dado_recebido_s,
209       distancia                    => distancia_bcd,
210       dist_mem                     => dist_mem,
211       dist_mais_sens               => dist_sens,
212       angulo                       => angulo_bcd_hex
213     );
214
215   M0: mux_4x1_n --VALOR DO DISPLAY HEX0
216     generic map(4)
217     port map(
218       -- Inputs
219       D0  => distancia_bcd(3 downto 0), --SEL_MUX=00 (distancia0)
220       D1  => dado_recebido_s(3 downto 0), --SEL_MUX=01 (DADO_RX1)
221       D2  => dist_mem(3 downto 0), --SEL_MUX=10 (dist_mem0)
222       D3  => angulo_bcd_hex(3 downto 0),--SEL_MUX=11 (angulo0)
223       SEL => sel_mux,
224       -- Output
225       MUX_OUT => m0_out
226     );
227
228   M1: mux_4x1_n --VALOR DO DISPLAY HEX1
229     generic map(4)
230     port map(
231       -- Inputs
232       D0  => distancia_bcd(7 downto 4),--SEL_MUX=00 (distancia1)
233       D1  => dado_recebido_s(7 downto 4), --SEL_MUX=01 (dado_RX1)
234       D2  => dist_mem(7 downto 4), --SEL_MUX=10 (dist_mem1)
235       D3  => angulo_bcd_hex(11 downto 8),--SEL_MUX=11 (angulo1)
236       SEL => sel_mux,
237       -- Output
238       MUX_OUT => m1_out
```

```
239     );
240
241   M2: mux_4x1_n --VALOR DO DISPLAY HEX2
242     generic map(4)
243     port map(
244       -- Inputs
245       D0  => distancia_bcd(11 downto 8),--SEL_MUX=00 (distancia2)
246       D1  => estado_rx, --SEL_MUX=01 (estado_rx)
247       D2  => dist_mem(11 downto 8), -- --SEL_MUX=10 (dist_mem2)
248       D3  => angulo_bcd_hex(19 downto 16),--SEL_MUX=11 (angulo2)
249       SEL => sel_mux,
250       -- Output
251       MUX_OUT => m2_out
252     );
253
254   M3: mux_4x1_n --VALOR DO DISPLAY HEX3
255     generic map(4)
256     port map(
257       -- Inputs
258       D0  => "0000",--SEL_MUX=00 a definir...
259       D1  => dist_sens(3 downto 0), --SEL_MUX=00 (DADO_TX0)
260       D2  => "0000", --SEL_MUX=00 (dado_tx1)
261       D3  => "0000",--SEL_MUX=11 a definir...
262       SEL => sel_mux,
263       -- Output
264       MUX_OUT => m3_out
265     );
266
267   M4: mux_4x1_n --VALOR DO DISPLAY HEX4
268     generic map(4)
269     port map(
270       -- Inputs
271       D0  => estado_hcsr,
272       D1  => dist_sens(7 downto 4), -- DADO_TX1
273       D2  => contagem_mux_4bits,
274       D3  => "0000",
275       SEL => sel_mux,
276       -- Output
277       MUX_OUT => m4_out
278     );
279
280   M5: mux_4x1_n --VALOR DO DISPLAY HEX5
281     generic map(4)
282     port map(
283       -- Inputs
284       D0  => X"0",
285       D1  => dist_sens(11 downto 8),
286       D2  => estado_tx_sistema_seguranca,
287       D3  => sistema_seguranca_estado,
288       SEL => sel_mux,
289       -- Output
290       MUX_OUT => m5_out
291     );
292
293   H0: hex7seg port map(hexa => m0_out, sseg => display0);
294   H1: hex7seg port map(hexa => m1_out, sseg => display1);
295   H2: hex7seg port map(hexa => m2_out, sseg => display2);
```

```
296   H3: hex7seg port map(hexa => m3_out, sseg => display3);
297   H4: hex7seg port map(hexa => m4_out, sseg => display4);
298   H5: hex7seg port map(hexa => m5_out, sseg => display5);
299
300   -- Outputs
301   pwm    <= pwm_s;
302   pwm_ch <= pwm_s;
303
304   trigger <= trigger_s;
305
306   calibrando <= calibrando_s;
307
308   saida_serial      <= saida_serial_s;
309   saida_serial_ch   <= saida_serial_s;
310   saida_serial_mqtt <= saida_serial_s;
311
312   alerta_proximidade <= alerta_proximidade_s;
313   alerta_prox_mqtt   <= alerta_proximidade_s;
314
315   db_medir   <= medir_s;
316   db_trigger <= trigger_s;
317   db_echo    <= echo;
318
319 end architecture;
```

8.1.2 Control Unit

```
1  library ieee;
2  use ieee.std_logic_1164.all;
3  use ieee.numeric_std.all;
4
5  entity sistema_seguranca_uc is
6      port (
7          -- Inputs
8          clock      : in std_logic;
9          reset      : in std_logic;
10         ligar      : in std_logic;
11         fim_1s     : in std_logic;
12         pronto_med : in std_logic;
13         pronto_tx  : in std_logic;
14         fim_cal    : in std_logic;
15       alerta     : in std_logic;
16         -- Outputs
17         zera       : out std_logic;
18         posiciona  : out std_logic;
19         medir      : out std_logic;
20         transmitir : out std_logic;
21         calibrando : out std_logic;
22         write_en   : out std_logic;
23       alerta_out : out std_logic;
24         db_estado  : out std_logic_vector(3 downto 0)
25     );
26 end entity;
27
28 architecture sistema_seguranca_uc_arch of sistema_seguranca_uc is
29
30     type tipo_estado is (idle, preparacao, cal_medida,
```

```
31                          cal_espera, cal_localizacao,
    cal_ultima_med,
32                          cal_armazena_ultima, medida, espera_med,
33                          transmissao, localizacao, analise,
    detectado);
34      signal Eatual : tipo_estado;
35      signal Eprox  : tipo_estado;
36
37  begin
38
39      -- Memoria de estado
40      process(clock, reset)
41      begin
42          if (reset = '1' or ligar = '0') then
43              Eatual <= idle;
44          elsif clock'event and clock = '1' then
45              Eatual <= Eprox;
46          end if;
47      end process;
48
49      -- Logica de proximo estado
50      process(ligar, fim_1s, pronto_med, pronto_tx, fim_cal, alerta,
        Eatual)
51      begin
52          case Eatual is
53              when idle =>
54                  if ligar = '1' then
55                      Eprox <= preparacao;
56                  else
57                      Eprox <= idle;
58                  end if;
59
60              when preparacao => Eprox <= cal_medida;
61
62              -- Calibracao
63              when cal_medida => Eprox <= cal_espera;
64
65              when cal_espera =>
66                  if fim_cal = '1' then
67                      Eprox <= cal_ultima_med;
68                  elsif pronto_med = '1' then
69                      Eprox <= cal_localizacao;
70                  elsif fim_1s = '1' then
71                      Eprox <= cal_localizacao;
72                  else
73                      Eprox <= cal_espera;
74                  end if;
75
76              when cal_localizacao =>
77                  if fim_1s = '1' then
78                      Eprox <= cal_medida;
79                  else
80                      Eprox <= cal_localizacao;
81                  end if;
82
83              when cal_ultima_med =>
84                  if pronto_med = '1' then
```

```
85                      Eprox <= cal_armazena_ultima;
86                  elsif fim_1s = '1' then
87                      Eprox <= cal_armazena_ultima;
88                  else
89                      Eprox <= cal_ultima_med;
90                  end if;
91
92              when cal_armazena_ultima => Eprox <= localizacao;
93
94              -- Deteccao de movimento
95              when medida => Eprox <= espera_med;
96
97              when espera_med =>
98                  if pronto_med = '1' then
99                      Eprox <= analise;
100                 elsif fim_1s = '1' then
101                     Eprox <= localizacao;
102                 else
103                     Eprox <= espera_med;
104                 end if;
105
106         when analise =>
107            if alerta = '1' then
108             Eprox <= transmissao;
109            else
110             Eprox <= localizacao;
111            end if;
112
113             when transmissao =>
114            if pronto_tx = '1' then
115             Eprox <= detectado;
116            else
117             Eprox <= transmissao;
118            end if;
119
120         when detectado => Eprox <= detectado;
121
122             when localizacao =>
123                 if fim_1s = '1' then
124                     Eprox <= medida;
125                 else
126                     Eprox <= localizacao;
127                 end if;
128
129             when others => Eprox <= idle;
130         end case;
131     end process;
132
133     -- Logica de saida
134     with Eatual select
135         zera <= '1' when preparacao, '0' when others;
136
137     with Eatual select
138         medir <= '1' when medida,
139                  '1' when cal_medida,
140                  '0' when others;
141
```

```
142     with Eatual select
143         transmitir <= '1' when transmissao, '0' when others;
144 
145     with Eatual select
146         posiciona <= '1' when localizacao,
147                      '1' when cal_localizacao,
148                      '0' when others;
149 
150     with Eatual select
151         calibrando <= '1' when cal_medida,
152                       '1' when cal_espera,
153                       '1' when cal_localizacao,
154                       '1' when cal_ultima_med,
155                       '1' when cal_armazena_ultima,
156                       '0' when others;
157 
158     with Eatual select
159         write_en <= '1' when cal_armazena_ultima, '0' when others;
160 
161    with Eatual select
162       alerta_out <= '1' when detectado, '0' when others;
163 
164     -- Debug
165     with Eatual select
166         db_estado <= "0000" when idle,
167                      "0001" when preparacao,
168                      "0010" when medida,
169                      "0011" when espera_med,
170                      "0100" when transmissao,
171                      "0101" when localizacao,
172                      "0110" when cal_medida,
173                      "0111" when cal_espera,
174                      "1000" when cal_localizacao,
175                      "1001" when cal_ultima_med,
176                      "1010" when cal_armazena_ultima,
177                "1011" when analise,
178                "1100" when detectado;
179 
180 end architecture;
```

8.1.3 Data Flow

```
1  library ieee;
2  use ieee.std_logic_1164.all;
3  use ieee.numeric_std.all;
4  use ieee.math_real.all;
5 
6  entity sistema_seguranca_fd is
7      port (
8      -- Inputs
9      clock       : in std_logic;
10     reset       : in std_logic;
11     ligar       : in std_logic;
12     medir       : in std_logic;
13     posiciona   : in std_logic;
14     transmitir  : in std_logic;
15     echo        : in std_logic;
```

```
16      dado_serial : in std_logic;
17      calibrando  : in std_logic;
18      write_en    : in std_logic;
19      -- Outputs
20      pwm                          : out std_logic;
21      trigger                      : out std_logic;
22      saida_serial                 : out std_logic;
23      pronto_tx                    : out std_logic;
24      alerta_proximidade           : out std_logic;
25      fim_1s                       : out std_logic;
26      meio_1s                      : out std_logic;
27      pronto_med                   : out std_logic;
28      fim_cal                      : out std_logic;
29      contagem_mux                 : out std_logic_vector(2 downto 0);
30      estado_hcsr                  : out std_logic_vector(3 downto 0);
31      estado_tx_sistema_seguranca : out std_logic_vector(3 downto 0);
32      estado_rx                    : out std_logic_vector(3 downto 0);
33      estado_tx                    : out std_logic_vector(3 downto 0);
34      posicao_servo                : out std_logic_vector(4 downto 0);
35      dado_recebido                : out std_logic_vector(7 downto 0);
36      distancia                    : out std_logic_vector(11 downto 0)
        ;
37      dist_mem                     : out std_logic_vector(11 downto 0)
        ;
38      dist_mais_sens               : out std_logic_vector(11 downto 0)
        ;
39      angulo                       : out std_logic_vector(23 downto 0)
40      );
41  end entity;
42
43  architecture sistema_seguranca_fd_arch of sistema_seguranca_fd is
44
45    -- Controle da movimenta  o do servo motor
46    component movimentacao_servomotor is
47      port (
48        -- Inputs
49        clock     : in std_logic;
50        reset     : in std_logic;
51        posiciona : in std_logic;
52        -- Outputs
53        pwm       : out std_logic;
54        fim_1s    : out std_logic;
55        meio_1s   : out std_logic;
56        last_pos  : out std_logic;
57        posicao   : out std_logic_vector(4 downto 0)
58      );
59    end component;
60
61    -- Transmissao Dados sistema_seguranca
62      component tx_dados is
63          port (
64        -- Inputs
65        clock      : in std_logic;
66        reset      : in std_logic;
67        transmitir : in std_logic;
68        dado_serial: in std_logic;
69        angulo2    : in std_logic_vector(3 downto 0);
```

```
70      angulo1    : in std_logic_vector(3 downto 0);
71      angulo0    : in std_logic_vector(3 downto 0);
72      distancia2 : in std_logic_vector(3 downto 0);
73      distancia1 : in std_logic_vector(3 downto 0);
74      distancia0 : in std_logic_vector(3 downto 0);
75      -- Outputs
76      saida_serial : out std_logic;
77      pronto       : out std_logic;
78      pronto_rx    : out std_logic;
79      contagem_mux : out std_logic_vector(2 downto 0);
80      dado_recebido: out std_logic_vector(7 downto 0);
81      -- Debug
82      db_transmitir   : out std_logic;
83      db_saida_serial : out std_logic;
84      db_estado_tx    : out std_logic_vector(3 downto 0);
85      db_estado_rx    : out std_logic_vector(3 downto 0)
86    );
87    end component;
88
89    -- Detetor de Borda
90    component edge_detector is
91        port (
92            -- Inputs
93            clk       : in   std_logic;
94            signal_in : in   std_logic;
95            -- Output
96            output : out  std_logic
97        );
98    end component;
99
100 -- Interface HCSR04
101 component interface_hcsr04
102   port (
103     -- Inputs
104     clock : in std_logic;
105     reset : in std_logic;
106     medir : in std_logic;
107     echo  : in std_logic;
108     -- Outputs
109     trigger : out std_logic;
110     pronto  : out std_logic;
111     medida  : out std_logic_vector(11 downto 0); -- 3 digitos BCD
112     -- Debug
113     db_estado : out std_logic_vector(3 downto 0)
114   );
115 end component;
116
117 component comparador_85 is
118   port (
119     i_A3   : in  std_logic; -- i_A3, i_A2, i_A1 e i_A0
120                 -- sao os bits que formam a palavra A
121     i_B3   : in  std_logic; -- i_B3, i_B2, i_B1 e i_B0
122                 -- sao os bits que formam a palavra B
123     i_A2   : in  std_logic;
124     i_B2   : in  std_logic;
125     i_A1   : in  std_logic;
126     i_B1   : in  std_logic;
```

```
127       i_A0   : in  std_logic;
128       i_B0   : in  std_logic;
129       i_AGTB : in  std_logic; -- i_AGTB, i_ALTB e i_AEQB sao inputs
130                    -- de cascateamento
131       i_ALTB : in  std_logic;
132       i_AEQB : in  std_logic;
133       o_AGTB : out std_logic; -- Saida em alto se A>B
134       o_ALTB : out std_logic; -- Saida em alto se A<B
135       o_AEQB : out std_logic  -- Saida em alto se A=B
136     );
137   end component;
138
139   -- RAM com distancias
140   component ram_dist is
141     port (
142       -- Inputs
143       clock : in std_logic;
144       wr    : in std_logic;
145       addr  : in std_logic_vector(4 downto 0);
146       din   : in std_logic_vector(11 downto 0);
147       -- Output
148       dout : out std_logic_vector(11 downto 0)
149     );
150   end component;
151
152   -- Registrador generico
153   component registrador_n is
154     generic (
155       constant N: integer := 8
156     );
157     port (
158       clock:  in  std_logic;
159       clear:  in  std_logic;
160       enable: in  std_logic;
161       D:      in  std_logic_vector (N-1 downto 0);
162       Q:      out std_logic_vector (N-1 downto 0)
163     );
164   end component;
165
166   -- Contador Generico
167     component contadorg_m is
168         generic (
169             constant M: integer := 50 -- modulo do contador
170         );
171         port (
172             -- Inputs
173             clock   : in std_logic;
174             zera_as : in std_logic;
175             zera_s  : in std_logic;
176             conta   : in std_logic;
177             -- Outputs
178             Q    : out std_logic_vector (natural(ceil(log2(real(M))
    ))-1 downto 0);
179             fim  : out std_logic;
180             meio : out std_logic
181         );
182     end component;
```

```
183
184   -- Full Adder
185   component fullAdder is
186     port(
187       -- Inputs
188       a, b, cin : in std_logic;
189       -- Outputs
190       s, cout   : out std_logic
191     );
192   end component;
193
194   -- Sinal
195   signal clear_reg        : std_logic;
196   signal pronto_med_s     : std_logic;
197   signal trigger_s        : std_logic;
198   signal write_stop       : std_logic;
199   signal write_enable     : std_logic;
200   signal last_pos         : std_logic;
201   signal last_pos_ed      : std_logic;
202   signal calibra          : std_logic;
203   signal agtb_vector      : std_logic_vector(3 downto 0);
204   signal altb_vector      : std_logic_vector(3 downto 0);
205   signal aeqb_vector      : std_logic_vector(3 downto 0);
206   signal altb_vector_reg  : std_logic_vector(3 downto 0);
207   signal posicao_s        : std_logic_vector(4 downto 0);
208   signal dado_recebido_s  : std_logic_vector(7 downto 0);
209   signal distancia_hcsr   : std_logic_vector(11 downto 0);
210   signal distancia_ram    : std_logic_vector(11 downto 0);
211   signal sensibilidade    : std_logic_vector(11 downto 0);
212   signal dist_sens        : std_logic_vector(11 downto 0);
213   signal carry            : std_logic_vector(12 downto 0);
214   signal angulo_rom       : std_logic_vector(23 downto 0);
215
216 begin
217
218   -- Logica de sinais
219   clear_reg    <= reset or posiciona;
220   calibra      <= calibrando and last_pos_ed;
221   write_enable <= (calibrando and (not write_stop)) or write_en;
222
223   -- Inicializacao
224   agtb_vector(0) <= '0';
225   altb_vector(0) <= '0';
226   aeqb_vector(0) <= '0';
227
228   carry(0) <= '1';
229
230   sensibilidade <= not B"0000_0000_0100";
231
232   -- Instancias
233     U1_TX: tx_dados
234         port map(
235             -- Inputs
236             clock       => clock,
237             reset       => reset,
238             transmitir  => transmitir,
239        dado_serial => dado_serial,
```

```
240             angulo2     => angulo_rom(19 downto 16),
241             angulo1     => angulo_rom(11 downto 8),
242             angulo0     => angulo_rom(3 downto 0),
243             distancia2  => distancia_hcsr(11 downto 8),
244             distancia1  => distancia_hcsr(7 downto 4),
245             distancia0  => distancia_hcsr(3 downto 0),
246             -- Outputs
247             saida_serial  => saida_serial,
248             pronto        => pronto_tx,
249       pronto_rx     => open,
250       contagem_mux  => contagem_mux,
251       dado_recebido => dado_recebido_s,
252             -- Debug
253             db_transmitir   => open,
254             db_saida_serial => open,
255             db_estado_tx    => estado_tx,
256       db_estado_rx    => estado_rx
257         );
258
259   U2_MOV: movimentacao_servomotor
260     port map(
261       -- Inputs
262       clock     => clock,
263       reset     => reset,
264       posiciona => posiciona,
265       -- Outputs
266       pwm       => pwm,
267       fim_1s    => fim_1s,
268       meio_1s   => meio_1s,
269       last_pos  => last_pos,
270       posicao   => posicao_s
271     );
272
273   U3_HCS: interface_hcsr04
274     port map(
275       -- Entradas
276       clock => clock,
277       reset => reset,
278       medir => medir,
279       echo  => echo,
280       -- Saidas
281       trigger => trigger,
282       pronto  => pronto_med_s,
283       medida  => distancia_hcsr, -- 3 digitos BCD
284       -- Debug
285       db_estado => estado_hcsr
286     );
287
288   GEN_COMP: for i in 0 to 2 generate
289     UX_CPX: comparador_85
290       port map(
291         -- Inputs
292         i_A3 => distancia_hcsr(4*i + 3),
293         i_B3 => dist_sens(4*i + 3),
294         i_A2 => distancia_hcsr(4*i + 2),
295         i_B2 => dist_sens(4*i + 2),
296         i_A1 => distancia_hcsr(4*i + 1),
```

```
297         i_B1 => dist_sens(4*i + 1),
298         i_A0 => distancia_hcsr(4*i),
299         i_B0 => dist_sens(4*i),
300         -- Cascateamento
301         i_AGTB => agtb_vector(i),
302         i_ALTB => altb_vector(i),
303         i_AEQB => aeqb_vector(i),
304         -- Outputs
305         o_AGTB => agtb_vector(i + 1),
306         o_ALTB => altb_vector(i + 1),
307         o_AEQB => aeqb_vector(i + 1)
308       );
309   end generate;
310
311   GEN_ADD: for i in 0 to 11 generate
312     UX_ADX: fullAdder
313       port map(
314         -- Inputs
315         a   => distancia_ram(i),
316         b   => sensibilidade(i),
317         cin => carry(i),
318         -- Outputs
319         s    => dist_sens(i),
320         cout => carry(i + 1)
321       );
322   end generate;
323
324   U7_REG: registrador_n
325     generic map(4)
326     port map(
327       clock  => clock,
328       clear  => clear_reg,
329       enable => pronto_med_s,
330       D      => altb_vector,
331       Q      => altb_vector_reg
332     );
333
334   U8_RAM: ram_dist
335     port map(
336       -- Inputs
337       clock => clock,
338       wr    => write_enable,
339       addr  => posicao_s,
340       din   => distancia_hcsr,
341       -- Output
342       dout => distancia_ram
343     );
344
345   -- Conta numero de varreduras: n + 1
346   -- Sendo 2, varre 2 + 1 = 3 vezes
347   U9_CONT: contadorg_m
348     generic map(2)
349     port map(
350       -- Inputs
351       clock   => clock,
352       zera_as => reset,
353       zera_s  => '0',
```

```
354       conta   => calibra,
355       -- Outputs
356       Q    => open,
357       fim  => write_stop,
358       meio => open
359     );
360
361   U10_ED: edge_detector
362     port map(
363       -- Inputs
364             clk       => clock,
365             signal_in => last_pos,
366             -- Output
367             output => last_pos_ed
368     );
369
370   -- Outputs
371   dado_recebido      <= dado_recebido_s;
372   pronto_med         <= pronto_med_s;
373   posicao_servo      <= posicao_s;
374   distancia          <= distancia_hcsr;
375   dist_mem           <= distancia_ram;
376   angulo             <= angulo_rom;
377   fim_cal            <= write_stop;
378   alerta_proximidade <= altb_vector_reg(3);
379   dist_mais_sens     <= dist_sens;
380
381 end architecture;
```

8.2 Sonar Data Flow Components

8.2.1 Single Port RAM

```
1  library ieee;
2  use ieee.std_logic_1164.all;
3  use ieee.numeric_std.all;
4  use ieee.math_real.all;
5
6  entity ram_dist is
7      port (
8          -- Inputs
9          clock : in std_logic;
10         wr    : in std_logic;
11         addr  : in std_logic_vector(4 downto 0);
12         din   : in std_logic_vector(11 downto 0);
13         -- Output
14         dout : out std_logic_vector(11 downto 0)
15     );
16 end entity;
17
18 architecture ram_dist_arch of ram_dist is
19
20     type ram_array is array (0 to 25) of std_logic_vector(11 downto
       0);
21     signal ram_block : ram_array;
22
```

```
23 begin
24
25     process(clock)
26     begin
27         if (clock'event and clock = '1') then
28             if wr = '1' then
29                 ram_block(to_integer(unsigned(addr))) <= din;
30             end if;
31
32             dout <= ram_block(to_integer(unsigned(addr)));
33         end if;
34     end process;
35
36 end architecture;
```

8.3 Testbench

8.3.1 Security System Testbench

```
1  library ieee;
2  use ieee.std_logic_1164.all;
3  use ieee.numeric_std.all;
4
5  entity sistema_seguranca_tb is
6  end entity;
7
8  architecture tb of sistema_seguranca_tb is
9      -- Inputs
10     signal clock_in       : std_logic := '0';
11     signal reset_in       : std_logic := '0';
12     signal ligar_in       : std_logic := '0';
13     signal echo_in        : std_logic := '0';
14     signal dado_serial_in : std_logic := '0';
15     signal sel_mux_in     : std_logic_vector(1 downto 0) := "00";
16
17     -- Outputs
18     signal trigger_out      : std_logic := '0';
19     signal pwm_out          : std_logic := '0';
20     signal saida_serial_out : std_logic := '0';
21     signal alerta_prox_out  : std_logic := '0';
22     signal calibrando_out   : std_logic := '0';
23     signal alerta_mov_out   : std_logic := '0';
24
25     -- Controle do clock
26     signal keep_simulating : std_logic := '0';
27     constant clockPeriod   : time      := 20 ns;
28
29     -- Array de casos de teste
30   type caso_teste_type is record
31     id    : natural;
32     tempo : integer;
33   end record;
34
35   type casos_teste_array is array (natural range <>) of
       caso_teste_type;
36   constant casos_teste : casos_teste_array :=
```

```
37      (
38        (1, 100),
39        (2, 231),
40              (3, 300),
41              (4, 392),
42              (5, 492),
43              (6, 392),
44              (7, 300),
45              (8, 231),
46              (9, 100),
47              (10, 231)
48      );
49
50    signal larguraPulso: time := 1 ns;
51
52  begin
53
54      -- Gerador de clock
55      clock_in <= (not clock_in) and keep_simulating after
        clockPeriod/2;
56
57      -- Instancia
58      DUT: entity work.sistema_seguranca (sistema_seguranca_arch)
59          port map(
60              -- Inputs
61              clock       => clock_in,
62              reset       => reset_in,
63              ligar       => ligar_in,
64              echo        => echo_in,
65              dado_serial => dado_serial_in,
66              sel_mux     => sel_mux_in,
67              -- Outputs
68              trigger            => trigger_out,
69              db_trigger         => open,
70              db_echo            => open,
71              pwm                => pwm_out,
72              saida_serial       => saida_serial_out,
73              saida_serial_ch    => open,
74              saida_serial_mqtt  => open,
75              pwm_ch             => open,
76              alerta_proximidade => alerta_prox_out,
77              alerta_prox_mqtt   => open,
78              db_transmitir      => open,
79              db_medir           => open,
80              calibrando         => calibrando_out,
81        alerta_mov => alerta_mov_out,
82              display0           => open,
83              display1           => open,
84              display2           => open,
85              display3           => open,
86              display4           => open,
87              display5           => open
88          );
89
90      -- Estimulo
91      stim: process is
92      begin
```

```
93          assert false report "Inicio das simulacoes" severity note;
94      keep_simulating <= '1';
95
96      ---- valores iniciais ----------------
97      echo_in  <= '0';
98
99      ---- inicio: reset ----------------
100     wait for 2*clockPeriod;
101     reset_in <= '1';
102     wait for 2 us;
103     reset_in <= '0';
104     wait until falling_edge(clock_in);
105
106     ---- espera de 100us
107     wait for 100 us;
108
109         wait until falling_edge(clock_in);
110         ligar_in <= '1';
111
112     ---- loop pelos casos de teste
113     for i in casos_teste'range loop
114       -- 1) determina largura do pulso echo
115       assert false report "Caso de teste " & integer'image(
    casos_teste(i).id) & ": " &
116         integer'image(casos_teste(i).tempo) & "us" severity note;
117       larguraPulso <= casos_teste(i).tempo * 1 us; -- caso de teste
     "i"
118
119       -- 2) espera por 400us (tempo entre trigger e echo)
120       wait for 400 us;
121
122       -- 3) gera pulso de echo (largura = larguraPulso)
123       echo_in <= '1';
124       wait for larguraPulso;
125       echo_in <= '0';
126
127       -- 4) espera entre casos de tese
128       wait for 20 ms;
129
130     end loop;
131
132     ---- final dos casos de teste da simulacao
133     assert false report "Fim das simulacoes" severity note;
134     keep_simulating <= '0';
135
136     wait; -- fim da simula  o: aguarda indefinidamente (n  o
    retirar esta linha)
137
138     end process;
139
140 end architecture;
```

Part III

Week 3: Modes of Use and Dashboard Processing

Chapter 9

Objectives

The objective of the second sprint of the project was to implement the operating modes of the security system – "at home" and "away from home". Additionally, corrections were made in relation to the development from week 2. Several considerations were made to correct the motion detection with the HC-SR04 sensor.

Chapter 10

Project Documentation

10.1 Considerations for Implementation

During the development of the last week, problems were encountered in implementing motion detection, as the sensor has some limitations due to being a low-cost alternative. The HC-SR04 component has difficulties in measuring spaces with corners, in measuring objects when the front part of the sensor is not perpendicular to the object, and in measuring objects that are too far away.

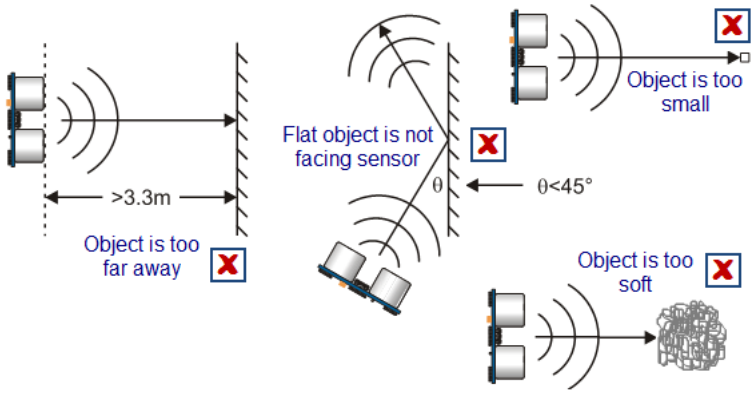

Figure 10.1: Graphical representation of the problems encountered when trying to measure objects with the HC-SR04 sensor.

In order to resolve this issue, some considerations were made to facilitate the use of a low-quality sensor. The room will be scanned using fewer positions with the servo motor, aiming to achieve $\theta > 45°$. During the demonstrations, an ideal environment will be used, which meets the requirements of objects and distance from 10.1.

10.2 Data Flow

The data flow underwent changes in the servo motor movement component. In previous weeks, there were 25 positions, which now have 11 positions. There was also a modification in the interval between measurements. Previously, the circuit performed a measurement every 1 second, but now it has been changed to a measurement every 2 seconds to prevent the circuit from freezing. Another change occurred in the distance counting. During the tests of the second week, it was noticed that the sensitivity of 4 *cm* was being subtracted from the BCD counter output. Therefore, this counter was replaced by a generic counter that does not perform the conversion for easy user readability – a modification that will be implemented.

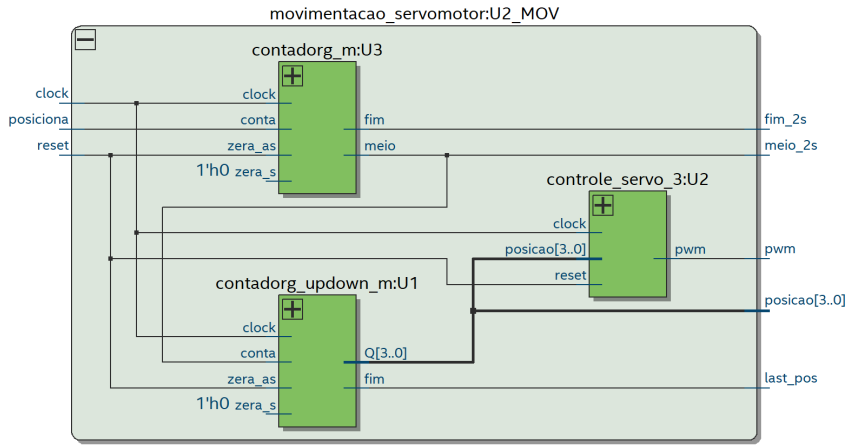

Figure 10.2: RTL view of the component that controls the movement of the servo motor. The output of the updown counter U1 now has a 4-bit vector as output, and the generic counter U3 counts up to 2 seconds indicating the end through the `fim_2s` signal.

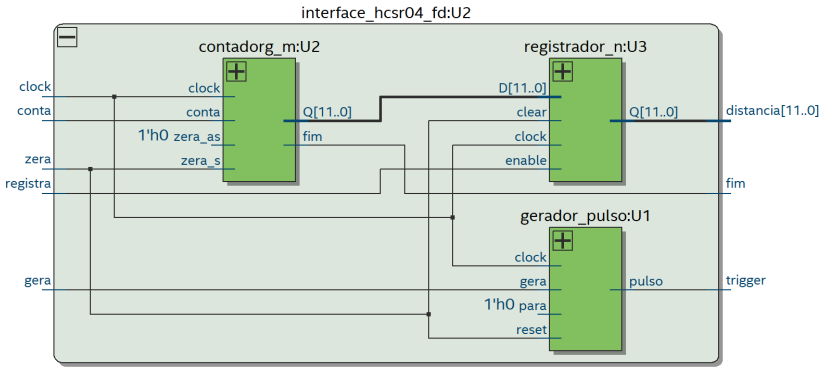

Figure 10.3: The BCD counter that was in the data flow of the HC-SR04 interface has been replaced by a generic counter that does not perform the conversion for easy user understanding.

10.3 State Machine

The change in the state machine was due to the implementation of the "at home" and "away from home" modes. Now there is a `std_logic` type input for selection. When the mode is "away from home" – input `mode` high –, when motion is detected, the circuit remains in the `detected` state and activates the circuit output `alerta_mov`. However, when the mode is "at home", the circuit activates the same output to identify that there is motion, but does not interrupt the circuit operation and only transmits the motion information to the `MQTT Dash`.

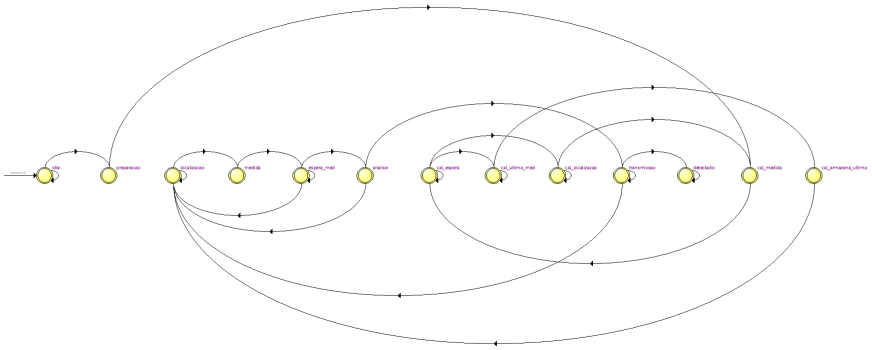

Figure 10.4: The number of states compared to the previous week has not changed, but the transitions have been modified to include the modes. Another modification is that now the transmission state is always used, not only when the circuit should interrupt operation.

	Source State	Destination State	Condition
1	analise	transmissao	(alerta)
2	analise	localizacao	(!alerta)
3	cal_armazena_ultima	localizacao	
4	cal_espera	cal_espera	(!pronto_med).(!fim_2s).(!fim_cal)
5	cal_espera	cal_localizacao	(!fim_2s).(pronto_med).(!fim_cal) + (fim_2s).(!fim_cal)
6	cal_espera	cal_ultima_med	(fim_cal)
7	cal_localizacao	cal_localizacao	(!fim_2s)
8	cal_localizacao	cal_medida	(fim_2s)
9	cal_medida	cal_espera	
10	cal_ultima_med	cal_armazena_ultima	(!fim_2s).(pronto_med) + (fim_2s)
11	cal_ultima_med	cal_ultima_med	(!fim_2s).(!pronto_med)
12	detectado	detectado	
13	espera_med	analise	(pronto_med).(!fim_2s)
14	espera_med	espera_med	(!pronto_med).(!fim_2s)
15	espera_med	localizacao	(fim_2s)
16	idle	preparacao	(ligar)
17	idle	idle	(!ligar)
18	localizacao	medida	(fim_2s)
19	localizacao	localizacao	(!fim_2s)
20	medida	espera_med	
21	preparacao	cal_medida	
22	transmissao	transmissao	(!pronto_tx)
23	transmissao	detectado	(pronto_tx).(mode)
24	transmissao	localizacao	(pronto_tx).(!mode)

Figure 10.5: State change logic. The `transmissao` state uses the `mode` input to define its destination.

Chapter 11

Tests

11.1 ModelSim Tests

Since simulating all positions and times used in the real scenario is costly, the described testbench reduces the number of servo motor positions to 5 and uses simulations of small distances. The goal is to validate the inclusion of both modes and observe if there has been any detriment to the functionalities implemented earlier.

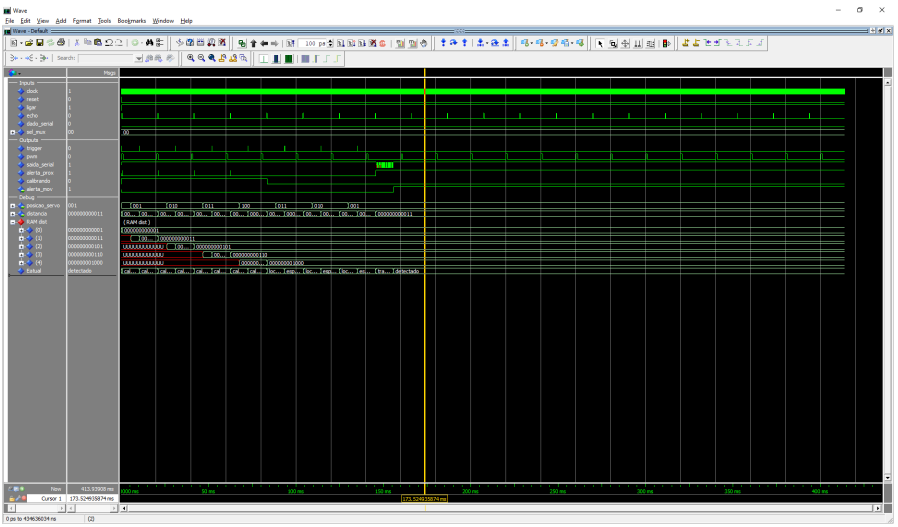

Figure 11.1: Simulation for the "away from home" mode, the first 5 measurements are for circuit calibration, these measurements are saved in the RAM `dist`, which has the outputs subtracted by 4 to be compared with the obtained measurements. When motion is detected, the circuit transmits the location and angle – will be changed later to the name of the room where it is installed – and interrupts the sweep.

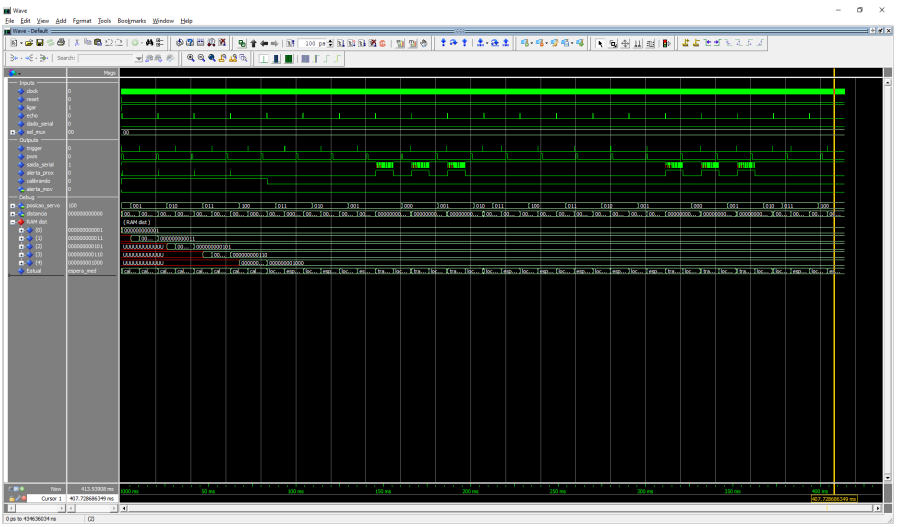

Figure 11.2: Simulation for the "at home" mode, the first 5 measurements are for circuit calibration, these measurements are saved in the RAM `dist`, which has the outputs subtracted by 4 to be compared with the obtained measurements. When motion is detected, the circuit transmits the location and angle – will be changed later to the name of the room where it is installed – and, in this mode, the sweep is not interrupted.

11.2 FPGA Tests

Using an ideal environment as proposed earlier in this document, it is possible to reduce the errors that the HC-SR04 sensor presents, but there is still a situation in which the sensor does not take any more measurements. In order to resolve this impasse, the code of the sensor interface developed in previous weeks is modified to include a logical input for the end of the timer. Thus, when the interface does not receive the echo and the interval for each measurement ends, the interface control unit returns to the initial state, after going through an exception state, to wait for the indication of a new measurement.

Part of the components needed to implement the password are already in the code, but commented. The password will be an integration with `Processing`.

Chapter 12

Processing Interface

With the aim of providing a graphical interface to the project users, the `Processing` tool presented in Experiment 6 of the discipline was used. Our `Processing` code interacts with the project through `MQTT`, publishing and receiving messages.

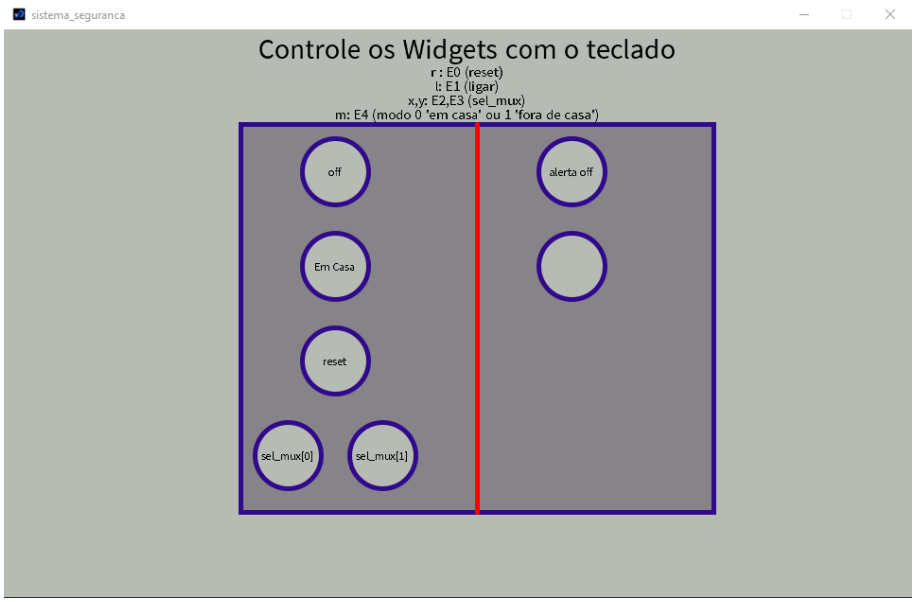

Figure 12.1: Overview of the interface built with `Processing` under the initial operating conditions of the circuit.

The interaction is done through the keyboard. When typing a certain key, a `MQTT` message is sent, changing the value of the corresponding signal, from 0 to 1 or from 1 to 0. In `MQTT` jargon, we act as `Publishers` of the input topics.

1. *reset*: controlled by the **r** key
2. *ligar*: controlled by the **l** key
3. *sel_mux[0]*: controlled by the **x** key
4. *sel_mux[1]*: controlled by the **y** key
5. *mode*: controlled by the **m** key

Besides operating the circuit and choosing debug signals, the interface also displays the values of two circuit outputs. In `MQTT` jargon, we act as `Subscribers` of the output topics.

1. *alerta_mov*
2. *calibrando*

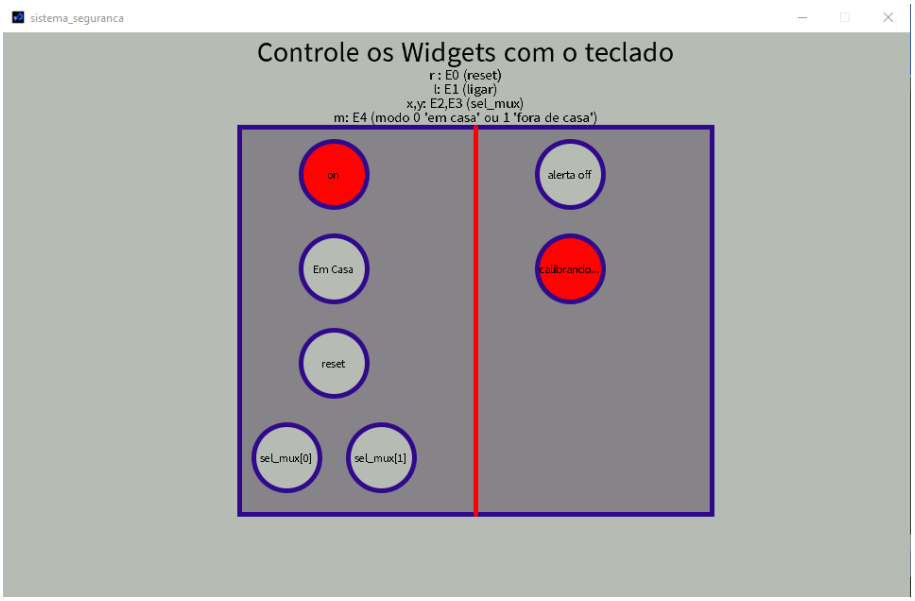

Figure 12.2: When activating the `ligar` input, we observe that the `calibrando` output is activated as observed in the circuit.

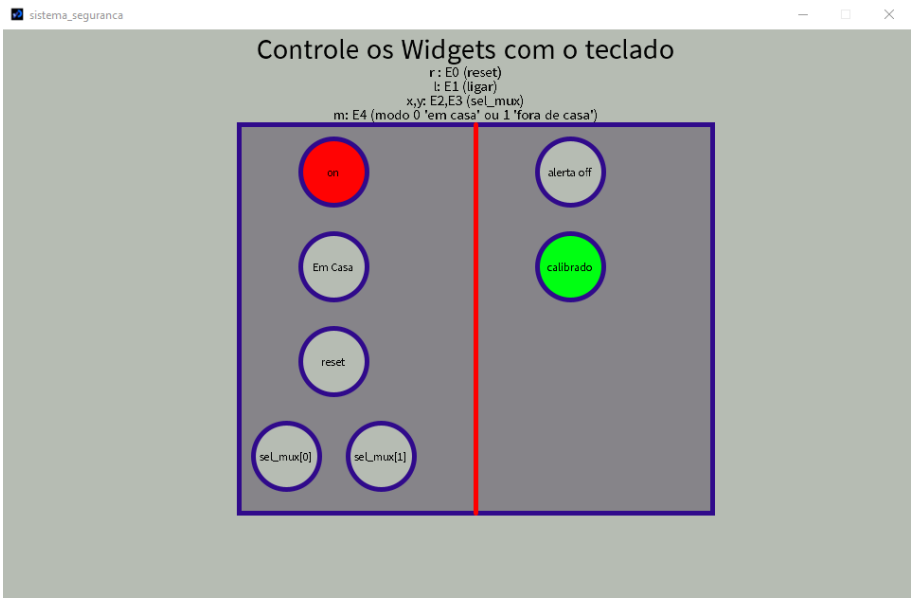

Figure 12.3: At the end of the calibration step, we see that again the `calibrando` output is deactivated.

If the motion alert `alerta_mov` is activated, a message is printed in the terminal indicating the time of detection. Later on, we will generate a logging file, which will store all these messages for user consultation.

```
if(Integer.parseInt(dados) == 1){//alerta esta ativado
  estado_alerta = true;
  DateTimeFormatter dtf = DateTimeFormatter.ofPattern("yyyy/MM/dd HH:mm:ss");
  LocalDateTime now = LocalDateTime.now();
  println("ALERTA DE MOVIMENTO : " + dtf.format(now));
}else{
  estado_alerta = false;
}
```

Figure 12.4: Portion of the code responsible for printing alert messages in the terminal.

The complete code is in the last chapter of this report and the `Processing` project `.zip` folder was submitted along with this report.

Chapter 13

Project Codes

13.1 Security System

13.1.1 Complete Circuit

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity sistema_seguranca is
  port(
    -- Inputs
    clock       : in std_logic;
    reset       : in std_logic;
    ligar       : in std_logic;
    echo        : in std_logic;
    dado_serial : in std_logic;
    mode        : in std_logic;
    sel_mux     : in std_logic_vector(1 downto 0);
    -- Outputs
    trigger             : out std_logic;
    db_trigger          : out std_logic;
    db_echo             : out std_logic;
    pwm                 : out std_logic;
    saida_serial        : out std_logic;
    saida_serial_ch     : out std_logic;
    saida_serial_mqtt   : out std_logic;
    pwm_ch              : out std_logic;
    alerta_proximidade  : out std_logic;
    alerta_prox_mqtt    : out std_logic;
    db_transmitir       : out std_logic;
    db_medir            : out std_logic;
    calibrando          : out std_logic;
    alerta_mov          : out std_logic;
    db_mode             : out std_logic;
    db_fim_2s           : out std_logic;
    display0            : out std_logic_vector(6 downto 0);
    display1            : out std_logic_vector(6 downto 0);
```

```
34      display2           : out std_logic_vector(6 downto 0);
35      display3           : out std_logic_vector(6 downto 0);
36      display4           : out std_logic_vector(6 downto 0);
37      display5           : out std_logic_vector(6 downto 0)
38    );
39  end entity;
40
41  architecture sistema_seguranca_arch of sistema_seguranca is
42
43    -- Unidade de Controle
44    component sistema_seguranca_uc is
45      port (
46        -- Inputs
47        clock      : in std_logic;
48        reset      : in std_logic;
49        ligar      : in std_logic;
50        fim_1s     : in std_logic;
51        pronto_med : in std_logic;
52        pronto_tx  : in std_logic;
53        fim_cal    : in std_logic;
54        alerta     : in std_logic;
55        mode       : in std_logic;
56        senha_ok   : in std_logic;
57        desarmar   : in std_logic;
58        -- Outputs
59        zera       : out std_logic;
60        posiciona  : out std_logic;
61        medir      : out std_logic;
62        transmitir : out std_logic;
63        calibrando : out std_logic;
64        write_en   : out std_logic;
65        alerta_out : out std_logic;
66        db_estado  : out std_logic_vector(3 downto 0)
67      );
68    end component;
69
70    -- Fluxo de Dados
71    component sistema_seguranca_fd is
72      port (
73        -- Inputs
74        clock       : in std_logic;
75        reset       : in std_logic;
76        ligar       : in std_logic;
77        medir       : in std_logic;
78        posiciona   : in std_logic;
79        transmitir  : in std_logic;
80        echo        : in std_logic;
81        dado_serial : in std_logic;
82        calibrando  : in std_logic;
83        write_en    : in std_logic;
84        alerta_mov  : in std_logic;
85        -- Outputs
86        pwm                       : out std_logic;
87        trigger                   : out std_logic;
88        saida_serial              : out std_logic;
89        pronto_tx                 : out std_logic;
90        alerta_proximidade        : out std_logic;
```

```
91      fim_1s                       : out std_logic;
92      meio_1s                      : out std_logic;
93      pronto_med                   : out std_logic;
94      fim_cal                      : out std_logic;
95      contagem_mux                 : out std_logic_vector(2 downto
    0);
96      estado_hcsr                  : out std_logic_vector(3 downto
    0);
97      estado_tx_sistema_seguranca : out std_logic_vector(3 downto
    0);
98      estado_rx                    : out std_logic_vector(3 downto
    0);
99      estado_tx                    : out std_logic_vector(3 downto
    0);
100     posicao_servo                : out std_logic_vector(3 downto
    0);
101     dado_recebido                : out std_logic_vector(7 downto
    0);
102     distancia                    : out std_logic_vector(11 downto
    0);
103     dist_mem                     : out std_logic_vector(11 downto
    0);
104     dist_mais_sens               : out std_logic_vector(11 downto
    0);
105     angulo                       : out std_logic_vector(23 downto
    0)
106   );
107 end component;
108
109 -- Multiplexador 4x1
110 component mux_4x1_n is
111   generic (
112     constant BITS: integer := 4
113   );
114   port (
115     D0 :     in  std_logic_vector (BITS-1 downto 0);
116     D1 :     in  std_logic_vector (BITS-1 downto 0);
117     D2 :     in  std_logic_vector (BITS-1 downto 0);
118     D3 :     in  std_logic_vector (BITS-1 downto 0);
119     SEL:     in  std_logic_vector (1 downto 0);
120     MUX_OUT: out std_logic_vector (BITS-1 downto 0)
121   );
122 end component;
123
124 -- Display de 7 segmentos
125 component hex7seg is
126   port (
127     hexa : in  std_logic_vector(3 downto 0);
128     sseg : out std_logic_vector(6 downto 0)
129   );
130 end component;
131
132 -- Sinais
133 signal pronto_tx_s                              : std_logic;
134 signal fim_1s_s, zera_s, reset_fd               : std_logic;
135 signal meio_1s_s                    : std_logic;
136 signal posiciona_s, medir_s                     : std_logic;
```

```
137   signal saida_serial_s, pwm_s                    : std_logic;
138   signal alerta_proximidade_s                     : std_logic;
139   signal transmitir_s, pronto_med_s               : std_logic;
140   signal fim_cal_s, calibrando_s, write_en_s      : std_logic;
141   signal trigger_s, mode_s, alerta_s              : std_logic;
142   signal contagem_mux_3bits                       :
        std_logic_vector(2 downto 0);
143   signal contagem_mux_4bits                       :
        std_logic_vector(3 downto 0);
144   signal sistema_seguranca_estado, posicao_4bits  :
        std_logic_vector(3 downto 0);
145   signal estado_hcsr, estado_tx_sistema_seguranca :
        std_logic_vector(3 downto 0);
146   signal estado_rx, estado_tx                     :
        std_logic_vector(3 downto 0);
147   signal dado_recebido_s                          :
        std_logic_vector(7 downto 0);
148   signal distancia_bcd                            :
        std_logic_vector(11 downto 0);
149   signal dist_mem                                 :
        std_logic_vector(11 downto 0);
150   signal dist_sens                                :
        std_logic_vector(11 downto 0);
151   signal angulo_bcd_hex                           :
        std_logic_vector(23 downto 0);
152
153   -- Saidas dos multiplexadores
154   signal m0_out, m1_out, m2_out, m3_out, m4_out, m5_out :
        std_logic_vector(3 downto 0);
155
156 begin
157
158   -- Logica de sinais
159   reset_fd           <= reset or zera_s;
160   mode_s             <= mode;
161   contagem_mux_4bits <= '0' & contagem_mux_3bits;
162
163   -- Instancias
164   U1_UC: sistema_seguranca_uc
165     port map(
166       -- Inputs
167       clock      => clock,
168       reset      => reset,
169       ligar      => ligar,
170       fim_1s     => fim_1s_s,
171       pronto_med => pronto_med_s,
172       pronto_tx  => pronto_tx_s,
173       fim_cal    => fim_cal_s,
174       alerta     => alerta_proximidade_s,
175       mode       => mode_s,
176       senha_ok   => '0',
177       desarmar   => '0',
178       -- Outputs
179       zera       => zera_s,
180       posiciona  => posiciona_s,
181       medir      => medir_s,
182       transmitir => transmitir_s,
```

```
183       calibrando => calibrando_s,
184       write_en   => write_en_s,
185       alerta_out => alerta_s,
186       db_estado  => sistema_seguranca_estado
187     );
188
189   U2_FD: sistema_seguranca_fd
190     port map(
191       -- Inputs
192       clock       => clock,
193       reset       => reset_fd,
194       ligar       => ligar,
195       medir       => medir_s,
196       posiciona   => posiciona_s,
197       transmitir  => transmitir_s,
198       echo        => echo,
199       dado_serial => dado_serial,
200       calibrando  => calibrando_s,
201       write_en    => write_en_s,
202       alerta_mov  => alerta_s,
203       -- Outputs
204       pwm                          => pwm_s,
205       trigger                      => trigger_s,
206       saida_serial                 => saida_serial_s,
207       pronto_tx                    => pronto_tx_s,
208       alerta_proximidade           => alerta_proximidade_s,
209       fim_1s                       => fim_1s_s,
210       meio_1s                      => meio_1s_s,
211       pronto_med                   => pronto_med_s,
212       fim_cal                      => fim_cal_s,
213       contagem_mux                 => contagem_mux_3bits,
214       estado_hcsr                  => estado_hcsr,
215       estado_tx_sistema_seguranca => estado_tx_sistema_seguranca,
216       estado_rx                    => estado_rx,
217       estado_tx                    => estado_tx,
218       posicao_servo                => posicao_4bits,
219       dado_recebido                => dado_recebido_s,
220       distancia                    => distancia_bcd,
221       dist_mem                     => dist_mem,
222       dist_mais_sens               => dist_sens,
223       angulo                       => angulo_bcd_hex
224     );
225
226   M0: mux_4x1_n --VALOR DO DISPLAY HEX0
227     generic map(4)
228     port map(
229       -- Inputs
230       D0  => distancia_bcd(3 downto 0), --SEL_MUX=00 (distancia0)
231       D1  => dado_recebido_s(3 downto 0), --SEL_MUX=01 (DADO_RX1)
232       D2  => dist_mem(3 downto 0), --SEL_MUX=10 (dist_mem0)
233       D3  => angulo_bcd_hex(3 downto 0),--SEL_MUX=11 (angulo0)
234       SEL => sel_mux,
235       -- Output
236       MUX_OUT => m0_out
237     );
238
239   M1: mux_4x1_n --VALOR DO DISPLAY HEX1
```

```
240     generic map(4)
241     port map(
242       -- Inputs
243       D0  => distancia_bcd(7 downto 4),--SEL_MUX=00 (distancia1)
244       D1  => dado_recebido_s(7 downto 4), --SEL_MUX=01 (dado_RX1)
245       D2  => dist_mem(7 downto 4), --SEL_MUX=10 (dist_mem1)
246       D3  => angulo_bcd_hex(11 downto 8),--SEL_MUX=11 (angulo1)
247       SEL => sel_mux,
248       -- Output
249       MUX_OUT => m1_out
250     );
251
252   M2: mux_4x1_n --VALOR DO DISPLAY HEX2
253     generic map(4)
254     port map(
255       -- Inputs
256       D0  => distancia_bcd(11 downto 8),--SEL_MUX=00 (distancia2)
257       D1  => estado_rx, --SEL_MUX=01 (estado_rx)
258       D2  => dist_mem(11 downto 8), -- --SEL_MUX=10 (dist_mem2)
259       D3  => angulo_bcd_hex(19 downto 16),--SEL_MUX=11 (angulo2)
260       SEL => sel_mux,
261       -- Output
262       MUX_OUT => m2_out
263     );
264
265   M3: mux_4x1_n --VALOR DO DISPLAY HEX3
266     generic map(4)
267     port map(
268       -- Inputs
269       D0  => "0000",--SEL_MUX=00 a definir...
270       D1  => dist_sens(3 downto 0), --SEL_MUX=00 (DADO_TX0)
271       D2  => "0000", --SEL_MUX=00 (dado_tx1)
272       D3  => "0000",--SEL_MUX=11 a definir...
273       SEL => sel_mux,
274       -- Output
275       MUX_OUT => m3_out
276     );
277
278   M4: mux_4x1_n --VALOR DO DISPLAY HEX4
279     generic map(4)
280     port map(
281       -- Inputs
282       D0  => estado_hcsr,
283       D1  => dist_sens(7 downto 4), -- DADO_TX1
284       D2  => contagem_mux_4bits,
285       D3  => "0000",
286       SEL => sel_mux,
287       -- Output
288       MUX_OUT => m4_out
289     );
290
291   M5: mux_4x1_n --VALOR DO DISPLAY HEX5
292     generic map(4)
293     port map(
294       -- Inputs
295       D0  => X"0",
296       D1  => dist_sens(11 downto 8),
```

```
297       D2  => estado_tx_sistema_seguranca,
298       D3  => sistema_seguranca_estado,
299       SEL => sel_mux,
300       -- Output
301       MUX_OUT => m5_out
302     );
303
304   H0: hex7seg port map(hexa => m0_out, sseg => display0);
305   H1: hex7seg port map(hexa => m1_out, sseg => display1);
306   H2: hex7seg port map(hexa => m2_out, sseg => display2);
307   H3: hex7seg port map(hexa => m3_out, sseg => display3);
308   H4: hex7seg port map(hexa => m4_out, sseg => display4);
309   H5: hex7seg port map(hexa => m5_out, sseg => display5);
310
311   -- Outputs
312   pwm    <= pwm_s;
313   pwm_ch <= pwm_s;
314
315   trigger <= trigger_s;
316
317   calibrando <= calibrando_s;
318
319   alerta_mov <= alerta_s;
320
321   saida_serial      <= saida_serial_s;
322   saida_serial_ch   <= saida_serial_s;
323   saida_serial_mqtt <= saida_serial_s;
324
325   alerta_proximidade <= alerta_proximidade_s;
326   alerta_prox_mqtt   <= alerta_proximidade_s;
327
328   db_medir   <= medir_s;
329   db_trigger <= trigger_s;
330   db_echo    <= echo;
331   db_mode    <= mode_s;
332   db_fim_2s  <= fim_1s_s;
333
334 end architecture;
```

13.1.2 Control Unit

```
1  library ieee;
2  use ieee.std_logic_1164.all;
3  use ieee.numeric_std.all;
4
5  entity sistema_seguranca_uc is
6      port (
7          -- Inputs
8          clock      : in std_logic;
9          reset      : in std_logic;
10         ligar      : in std_logic;
11         fim_1s     : in std_logic;
12         pronto_med : in std_logic;
13         pronto_tx  : in std_logic;
14         fim_cal    : in std_logic;
15         alerta     : in std_logic;
16         mode       : in std_logic;
```

```
17          senha_ok   : in std_logic;
18          desarmar   : in std_logic;
19          -- Outputs
20          zera       : out std_logic;
21          posiciona  : out std_logic;
22          medir      : out std_logic;
23          transmitir : out std_logic;
24          calibrando : out std_logic;
25          write_en   : out std_logic;
26          alerta_out : out std_logic;
27          db_estado  : out std_logic_vector(3 downto 0)
28      );
29  end entity;
30
31  architecture sistema_seguranca_uc_arch of sistema_seguranca_uc is
32
33      type tipo_estado is
34          (
35              idle, preparacao, cal_medida,
36              cal_espera, cal_localizacao, cal_ultima_med,
37              cal_armazena_ultima, medida, espera_med,
38              transmissao, localizacao, analise, detectado
39              -- ligar_auth, desarmar_auth
40          );
41
42      signal Eatual : tipo_estado;
43      signal Eprox  : tipo_estado;
44
45  begin
46
47      -- Memoria de estado
48      process(clock, reset)
49      begin
50          if (reset = '1' or ligar = '0') then
51              Eatual <= idle;
52          -- elsif ligar'event and event = '0' then
53          --     Eatual <= desarmar_auth;
54          elsif clock'event and clock = '1' then
55              Eatual <= Eprox;
56          end if;
57      end process;
58
59      -- Logica de proximo estado
60      process(ligar, fim_1s, pronto_med, pronto_tx, fim_cal, alerta,
        mode, senha_ok, desarmar, Eatual)
61      begin
62          case Eatual is
63              when idle =>
64                  if ligar = '1' then
65                      Eprox <= preparacao;
66                  else
67                      Eprox <= idle;
68                  end if;
69
70              when preparacao => Eprox <= cal_medida;
71
72              -- when preparacao => Eprox <= ligar_auth;
```

```
73
74              -- -- Armar
75              -- when ligar_auth =>
76              --     if senha_ok = '1' then
77              --         Eprox <= cal_medida;
78              --     else
79              --         Eprox <= ligar_auth;
80              --     end if;
81
82              -- Calibracao
83              when cal_medida => Eprox <= cal_espera;
84
85              when cal_espera =>
86                  if fim_cal = '1' then
87                      Eprox <= cal_ultima_med;
88                  elsif fim_1s = '1' then
89                      Eprox <= cal_localizacao;
90                  elsif pronto_med = '1' then
91                      Eprox <= cal_localizacao;
92                  else
93                      Eprox <= cal_espera;
94                  end if;
95
96              when cal_localizacao =>
97                  if fim_1s = '1' then
98                      Eprox <= cal_medida;
99                  else
100                     Eprox <= cal_localizacao;
101                 end if;
102
103             when cal_ultima_med =>
104                 if pronto_med = '1' then
105                     Eprox <= cal_armazena_ultima;
106                 elsif fim_1s = '1' then
107                     Eprox <= cal_armazena_ultima;
108                 else
109                     Eprox <= cal_ultima_med;
110                 end if;
111
112             when cal_armazena_ultima => Eprox <= localizacao;
113
114             -- Deteccao de movimento
115             when medida => Eprox <= espera_med;
116
117             when espera_med =>
118                 if fim_1s = '1' then
119                     Eprox <= medida;
120                 elsif pronto_med = '1' then
121                     Eprox <= analise;
122                 else
123                     Eprox <= espera_med;
124                 end if;
125
126             when analise =>
127                 if alerta = '1' then
128                     Eprox <= transmissao;
129                 else
```

```
130                     Eprox <= localizacao;
131                 end if;
132
133             when transmissao =>
134                 if pronto_tx = '1' and mode = '1' then
135                     Eprox <= detectado;
136                 elsif pronto_tx = '1' and mode = '0' then
137                     Eprox <= localizacao;
138                 else
139                     Eprox <= transmissao;
140                 end if;
141
142             when detectado => Eprox <= detectado;
143
144             -- when detectado =>
145             --     if desarmar = '1' then
146             --         Eprox <= desarmar_auth;
147             --     else
148             --         Eprox <= detectado;
149             --     end if;
150
151             when localizacao =>
152                 if fim_1s = '1' then
153                     Eprox <= medida;
154                 else
155                     Eprox <= localizacao;
156                 end if;
157
158             -- -- Desarmar
159             -- when desarmar_auth =>
160             --     if senha_ok = '1' then
161             --         Eprox <= idle;
162             --     else
163             --         Eprox <= desarmar_auth;
164             --     end if;
165
166             when others => Eprox <= idle;
167         end case;
168     end process;
169
170     -- Logica de saida
171     with Eatual select
172         zera <= '1' when preparacao, '0' when others;
173
174     with Eatual select
175         medir <=
176             '1' when medida,
177             '1' when cal_medida,
178             '0' when others;
179
180     with Eatual select
181         transmitir <= '1' when transmissao, '0' when others;
182
183     with Eatual select
184         posiciona <=
185             '1' when localizacao,
186             '1' when cal_localizacao,
```

```
187             '0' when others;
188
189     with Eatual select
190         calibrando <=
191             '1' when cal_medida,
192             '1' when cal_espera,
193             '1' when cal_localizacao,
194             '1' when cal_ultima_med,
195             '1' when cal_armazena_ultima,
196             '0' when others;
197
198     with Eatual select
199         write_en <= '1' when cal_armazena_ultima, '0' when others;
200
201    with Eatual select
202         alerta_out <= '1' when detectado, '0' when others;
203
204     -- Debug
205     with Eatual select
206         db_estado <=
207             "0000" when idle,
208             "0001" when preparacao,
209             "0010" when medida,
210             "0011" when espera_med,
211             "0100" when transmissao,
212             "0101" when localizacao,
213             "0110" when cal_medida,
214             "0111" when cal_espera,
215             "1000" when cal_localizacao,
216             "1001" when cal_ultima_med,
217             "1010" when cal_armazena_ultima,
218             "1011" when analise,
219             "1101" when detectado;
220             -- "1110" when armar_auth,
221             -- "1111" when desarmar_auth;
222
223 end architecture;
```

13.1.3 Data Flow

```
1  library ieee;
2  use ieee.std_logic_1164.all;
3  use ieee.numeric_std.all;
4  use ieee.math_real.all;
5
6  entity sistema_seguranca_fd is
7      port (
8      -- Inputs
9      clock       : in std_logic;
10     reset       : in std_logic;
11     ligar       : in std_logic;
12     medir       : in std_logic;
13     posiciona   : in std_logic;
14     transmitir  : in std_logic;
15     echo        : in std_logic;
16     dado_serial : in std_logic;
17     calibrando  : in std_logic;
```

```
18      write_en     : in std_logic;
19      alerta_mov   : in std_logic;
20      -- Outputs
21      pwm                          : out std_logic;
22      trigger                      : out std_logic;
23      saida_serial                 : out std_logic;
24      pronto_tx                    : out std_logic;
25      alerta_proximidade           : out std_logic;
26      fim_1s                       : out std_logic;
27      meio_1s                      : out std_logic;
28      pronto_med                   : out std_logic;
29      fim_cal                      : out std_logic;
30      contagem_mux                 : out std_logic_vector(2 downto 0);
31      estado_hcsr                  : out std_logic_vector(3 downto 0);
32      estado_tx_sistema_seguranca  : out std_logic_vector(3 downto 0);
33      estado_rx                    : out std_logic_vector(3 downto 0);
34      estado_tx                    : out std_logic_vector(3 downto 0);
35      posicao_servo                : out std_logic_vector(3 downto 0);
36      dado_recebido                : out std_logic_vector(7 downto 0);
37      distancia                    : out std_logic_vector(11 downto 0)
        ;
38      dist_mem                     : out std_logic_vector(11 downto 0)
        ;
39      dist_mais_sens               : out std_logic_vector(11 downto 0)
        ;
40      angulo                       : out std_logic_vector(23 downto 0)
41      );
42  end entity;
43
44  architecture sistema_seguranca_fd_arch of sistema_seguranca_fd is
45
46    -- Controle da movimenta  o do servo motor
47    component movimentacao_servomotor is
48      port (
49        -- Inputs
50        clock : in std_logic;
51        reset : in std_logic;
52        ligar : in std_logic;
53        -- Outputs
54        pwm      : out std_logic;
55        fim_1s   : out std_logic;
56        meio_1s  : out std_logic;
57        last_pos : out std_logic;
58        posicao  : out std_logic_vector(3 downto 0)
59      );
60    end component;
61
62    -- Transmissao Dados sistema_seguranca
63      component tx_dados is
64          port (
65        -- Inputs
66        clock      : in std_logic;
67        reset      : in std_logic;
68        transmitir : in std_logic;
69        dado_serial: in std_logic;
70        angulo2    : in std_logic_vector(3 downto 0);
71        angulo1    : in std_logic_vector(3 downto 0);
```

```
72      angulo0    : in std_logic_vector(3 downto 0);
73      distancia2 : in std_logic_vector(3 downto 0);
74      distancia1 : in std_logic_vector(3 downto 0);
75      distancia0 : in std_logic_vector(3 downto 0);
76      -- Outputs
77      saida_serial : out std_logic;
78      pronto       : out std_logic;
79      pronto_rx    : out std_logic;
80      contagem_mux : out std_logic_vector(2 downto 0);
81      dado_recebido: out std_logic_vector(7 downto 0);
82      -- Debug
83      db_transmitir   : out std_logic;
84      db_saida_serial : out std_logic;
85      db_estado_tx    : out std_logic_vector(3 downto 0);
86      db_estado_rx    : out std_logic_vector(3 downto 0)
87    );
88    end component;
89
90    -- Detetor de Borda
91    component edge_detector is
92        port (
93            -- Inputs
94            clk       : in   std_logic;
95            signal_in : in   std_logic;
96            -- Output
97            output : out  std_logic
98        );
99    end component;
100
101 -- Interface HCSR04
102 component interface_hcsr04
103   port (
104     -- Inputs
105     clock : in std_logic;
106     reset : in std_logic;
107     medir : in std_logic;
108     echo  : in std_logic;
109     timer : in std_logic;
110     -- Outputs
111     trigger : out std_logic;
112     pronto  : out std_logic;
113     medida  : out std_logic_vector(11 downto 0); -- 3 digitos BCD
114     -- Debug
115     db_estado : out std_logic_vector(3 downto 0)
116   );
117 end component;
118
119 component comparador_85 is
120   port (
121     i_A3   : in  std_logic; -- i_A3, i_A2, i_A1 e i_A0
122                 -- sao os bits que formam a palavra A
123     i_B3   : in  std_logic; -- i_B3, i_B2, i_B1 e i_B0
124                 -- sao os bits que formam a palavra B
125     i_A2   : in  std_logic;
126     i_B2   : in  std_logic;
127     i_A1   : in  std_logic;
128     i_B1   : in  std_logic;
```

```
129       i_A0   : in  std_logic;
130       i_B0   : in  std_logic;
131       i_AGTB : in  std_logic; -- i_AGTB, i_ALTB e i_AEQB sao inputs
132                    -- de cascateamento
133       i_ALTB : in  std_logic;
134       i_AEQB : in  std_logic;
135       o_AGTB : out std_logic; -- Saida em alto se A>B
136       o_ALTB : out std_logic; -- Saida em alto se A<B
137       o_AEQB : out std_logic  -- Saida em alto se A=B
138     );
139   end component;
140
141   -- RAM com distancias
142   component ram_dist is
143     port (
144       -- Inputs
145       clock : in std_logic;
146       wr    : in std_logic;
147       addr  : in std_logic_vector(3 downto 0);
148       din   : in std_logic_vector(11 downto 0);
149       -- Output
150       dout : out std_logic_vector(11 downto 0)
151     );
152   end component;
153
154   -- Registrador generico
155   component registrador_n is
156     generic (
157       constant N: integer := 8
158     );
159     port (
160       clock:  in  std_logic;
161       clear:  in  std_logic;
162       enable: in  std_logic;
163       D:      in  std_logic_vector (N-1 downto 0);
164       Q:      out std_logic_vector (N-1 downto 0)
165     );
166   end component;
167
168   -- Contador Generico
169     component contadorg_m is
170         generic (
171             constant M: integer := 50 -- modulo do contador
172         );
173         port (
174             -- Inputs
175             clock   : in std_logic;
176             zera_as : in std_logic;
177             zera_s  : in std_logic;
178             conta   : in std_logic;
179             -- Outputs
180             Q    : out std_logic_vector (natural(ceil(log2(real(M))
    ))-1 downto 0);
181             fim  : out std_logic;
182             meio : out std_logic
183         );
184     end component;
```

```
185
186   -- Full Adder
187   component fullAdder is
188     port(
189       -- Inputs
190       a, b, cin : in std_logic;
191       -- Outputs
192       s, cout   : out std_logic
193     );
194   end component;
195
196   -- Sinal
197   signal clear_reg        : std_logic;
198   signal pronto_med_s     : std_logic;
199   signal trigger_s        : std_logic;
200   signal write_stop       : std_logic;
201   signal write_enable     : std_logic;
202   signal last_pos         : std_logic;
203   signal last_pos_ed      : std_logic;
204   signal calibra          : std_logic;
205   signal ligar_servo      : std_logic;
206   signal fim_1s_s         : std_logic;
207   signal agtb_vector      : std_logic_vector(3 downto 0);
208   signal altb_vector      : std_logic_vector(3 downto 0);
209   signal aeqb_vector      : std_logic_vector(3 downto 0);
210   signal altb_vector_reg  : std_logic_vector(3 downto 0);
211   signal posicao_s        : std_logic_vector(3 downto 0);
212   signal dado_recebido_s  : std_logic_vector(7 downto 0);
213   signal distancia_hcsr   : std_logic_vector(11 downto 0);
214   signal distancia_ram    : std_logic_vector(11 downto 0);
215   signal sensibilidade    : std_logic_vector(11 downto 0);
216   signal dist_sens        : std_logic_vector(11 downto 0);
217   signal carry            : std_logic_vector(12 downto 0);
218   signal angulo_rom       : std_logic_vector(23 downto 0);
219
220 begin
221
222   -- Logica de sinais
223   clear_reg    <= reset or posiciona;
224   calibra      <= calibrando and last_pos_ed;
225   write_enable <= (calibrando and (not write_stop)) or write_en;
226   ligar_servo  <= ligar and (not alerta_mov);
227
228   -- Inicializacao
229   agtb_vector(0) <= '0';
230   altb_vector(0) <= '0';
231   aeqb_vector(0) <= '1';
232
233   carry(0) <= '1';
234
235   sensibilidade <= not B"0000_0000_0010";
236
237   -- Instancias
238     U1_TX: tx_dados
239         port map(
240             -- Inputs
241             clock       => clock,
```

```
242             reset       => reset,
243             transmitir  => transmitir,
244       dado_serial => dado_serial,
245             angulo2     => angulo_rom(19 downto 16),
246             angulo1     => angulo_rom(11 downto 8),
247             angulo0     => angulo_rom(3 downto 0),
248             distancia2  => distancia_hcsr(11 downto 8),
249             distancia1  => distancia_hcsr(7 downto 4),
250             distancia0  => distancia_hcsr(3 downto 0),
251             -- Outputs
252             saida_serial  => saida_serial,
253             pronto        => pronto_tx,
254       pronto_rx     => open,
255       contagem_mux  => contagem_mux,
256       dado_recebido => dado_recebido_s,
257             -- Debug
258             db_transmitir   => open,
259             db_saida_serial => open,
260             db_estado_tx    => estado_tx,
261       db_estado_rx    => estado_rx
262         );
263
264   U2_MOV: movimentacao_servomotor
265     port map(
266       -- Inputs
267       clock => clock,
268       reset => reset,
269       ligar => ligar_servo,
270       -- Outputs
271       pwm      => pwm,
272       fim_1s   => fim_1s_s,
273       meio_1s  => meio_1s,
274       last_pos => last_pos,
275       posicao  => posicao_s
276     );
277
278   U3_HCS: interface_hcsr04
279     port map(
280       -- Entradas
281       clock => clock,
282       reset => reset,
283       medir => medir,
284       echo  => echo,
285       timer => fim_1s_s,
286       -- Saidas
287       trigger => trigger,
288       pronto  => pronto_med_s,
289       medida  => distancia_hcsr, -- 3 digitos BCD
290       -- Debug
291       db_estado => estado_hcsr
292     );
293
294   GEN_COMP: for i in 0 to 2 generate
295     UX_CPX: comparador_85
296       port map(
297         -- Inputs
298         i_A3 => distancia_hcsr(4*i + 3),
```

```
299         i_B3 => dist_sens(4*i + 3),
300         i_A2 => distancia_hcsr(4*i + 2),
301         i_B2 => dist_sens(4*i + 2),
302         i_A1 => distancia_hcsr(4*i + 1),
303         i_B1 => dist_sens(4*i + 1),
304         i_A0 => distancia_hcsr(4*i),
305         i_B0 => dist_sens(4*i),
306         -- Cascateamento
307         i_AGTB => agtb_vector(i),
308         i_ALTB => altb_vector(i),
309         i_AEQB => aeqb_vector(i),
310         -- Outputs
311         o_AGTB => agtb_vector(i + 1),
312         o_ALTB => altb_vector(i + 1),
313         o_AEQB => aeqb_vector(i + 1)
314       );
315   end generate;
316
317   GEN_ADD: for i in 0 to 11 generate
318     UX_ADX: fullAdder
319       port map(
320         -- Inputs
321         a   => distancia_ram(i),
322         b   => sensibilidade(i),
323         cin => carry(i),
324         -- Outputs
325         s    => dist_sens(i),
326         cout => carry(i + 1)
327       );
328   end generate;
329
330   U7_REG: registrador_n
331     generic map(4)
332     port map(
333       clock  => clock,
334       clear  => clear_reg,
335       enable => pronto_med_s,
336       D      => altb_vector,
337       Q      => altb_vector_reg
338     );
339
340   U8_RAM: ram_dist
341     port map(
342       -- Inputs
343       clock => clock,
344       wr    => write_enable,
345       addr  => posicao_s,
346       din   => distancia_hcsr,
347       -- Output
348       dout => distancia_ram
349     );
350
351   -- Conta numero de varreduras: n + 1
352   -- Sendo 2, varre 2 + 1 = 3 vezes
353   U9_CONT: contadorg_m
354     generic map(2)
355     port map(
```

```
356       -- Inputs
357       clock   => clock,
358       zera_as => reset,
359       zera_s  => '0',
360       conta   => calibra,
361       -- Outputs
362       Q    => open,
363       fim  => write_stop,
364       meio => open
365     );
366
367   U10_ED: edge_detector
368     port map(
369       -- Inputs
370             clk       => clock,
371             signal_in => last_pos,
372             -- Output
373             output => last_pos_ed
374     );
375
376   -- Outputs
377   fim_1s             <= fim_1s_s;
378   dado_recebido      <= dado_recebido_s;
379   pronto_med         <= pronto_med_s;
380   posicao_servo      <= posicao_s;
381   distancia          <= distancia_hcsr;
382   dist_mem           <= distancia_ram;
383   angulo             <= angulo_rom;
384   fim_cal            <= write_stop;
385   alerta_proximidade <= altb_vector_reg(3);
386   dist_mais_sens     <= dist_sens;
387
388 end architecture;
```

13.2 Modified Data Flow Components

13.2.1 Servo Motor Movement

```
1  library ieee;
2  use ieee.std_logic_1164.all;
3  use ieee.numeric_std.all;
4  use ieee.math_real.all;
5
6  entity movimentacao_servomotor is
7      port (
8          -- Inputs
9          clock : in std_logic;
10         reset : in std_logic;
11         ligar : in std_logic;
12         -- Outputs
13         pwm      : out std_logic;
14         fim_1s   : out std_logic;
15         meio_1s  : out std_logic;
16         last_pos : out std_logic;
17         posicao  : out std_logic_vector(3 downto 0)
18     );
```

```
19 end entity;
20
21 architecture mov_servo_arch of movimentacao_servomotor is
22
23     -- Contador Up Down Generico
24     component contadorg_updown_m is
25         generic (
26             constant M: integer := 50 -- modulo do contador
27         );
28         port (
29             clock:   in  std_logic;
30             zera_as: in  std_logic;
31             zera_s:  in  std_logic;
32             conta:   in  std_logic;
33             Q:       out std_logic_vector (natural(ceil(log2(real(M
    ))))-1 downto 0);
34             inicio:  out std_logic;
35             fim:     out std_logic;
36             meio:    out std_logic
37         );
38     end component;
39
40     -- Controle Servo Revisado
41     component controle_servo_3 is
42         port (
43             -- Inputs
44             clock   : in  std_logic;
45             reset   : in  std_logic;
46             posicao : in  std_logic_vector(3 downto 0);
47             -- Output
48             pwm : out std_logic;
49             -- Debug
50             db_reset   : out std_logic;
51             db_pwm     : out std_logic;
52             db_posicao : out std_logic_vector(3 downto 0)
53         );
54     end component;
55
56     -- Contador Generico
57     component contadorg_m is
58         generic (
59             constant M: integer := 50 -- modulo do contador
60         );
61         port (
62             -- Inputs
63             clock   : in std_logic;
64             zera_as : in std_logic;
65             zera_s  : in std_logic;
66             conta   : in std_logic;
67             -- Outputs
68             Q    : out std_logic_vector (natural(ceil(log2(real(M))
    ))-1 downto 0);
69             fim  : out std_logic;
70             meio : out std_logic
71         );
72     end component;
73
```

```
74      -- Sinais
75      signal conta_updown   : std_logic;
76      signal tick, off_tick : std_logic;
77      signal pwm_s          : std_logic;
78      signal posicao_s      : std_logic_vector(3 downto 0);
79
80  begin
81      -- Instancias
82      U1: contadorg_updown_m
83          generic map(11)
84          port map(
85              -- Inputs
86              clock   => clock,
87              zera_as => reset,
88              zera_s  => '0',
89              conta   => off_tick,
90              -- Outputs
91              Q      => posicao_s,
92              inicio => open,
93              fim    => last_pos,
94              meio   => open
95          );
96
97      U2: controle_servo_3
98          port map(
99              -- Inputs
100             clock   => clock,
101             reset   => reset,
102             posicao => posicao_s,
103             -- Output
104             pwm => pwm,
105             -- Debug
106             db_reset   => open,
107             db_pwm     => open,
108             db_posicao => open
109         );
110
111     U3: contadorg_m
112         generic map(50000000) -- 1s = 50000000; 2s = 100000000
113         port map(
114             -- Inputs
115             clock   => clock,
116             zera_as => reset,
117             zera_s  => '0',
118             conta   => ligar,
119             -- Outputs
120             Q    => open,
121             fim  => tick,
122             meio => off_tick
123         );
124
125     -- Output
126     posicao <= posicao_s;
127     fim_1s  <= tick;
128     meio_1s <= off_tick;
129
130 end architecture;
```

13.2.2 HC-SR04 Interface Data Flow

```
library ieee;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;
use IEEE.math_real.all;

entity interface_hcsr04_fd is
    port (
        -- Entradas
        clock    : in std_logic;
        zera     : in std_logic;
        conta    : in std_logic;
        registra : in std_logic;
        gera     : in std_logic;
        -- Saidas
        trigger   : out std_logic;
        fim       : out std_logic;
        distancia : out std_logic_vector(11 downto 0)
    );
end entity;

architecture hcsr04_fd_arch of interface_hcsr04_fd is

    -- GERADOR DE PULSO
    component gerador_pulso is
        generic (
            largura: integer:= 25
        );
        port(
            clock:  in  std_logic;
            reset:  in  std_logic;
            gera:   in  std_logic;
            para:   in  std_logic;
            pulso:  out std_logic;
            pronto: out std_logic
        );
    end component;

    -- CONTADOR BCD DE 4 DIGITOS
    component contador_bcd_4digitos is
        port (clock, zera, conta:     in  std_logic;
              dig3, dig2, dig1, dig0: out std_logic_vector(3 downto
 0);
              fim:                    out std_logic
        );
    end component;

    --CONTADOR BINARIO
    component contadorg_m
        generic (
            constant M: integer := 50 -- modulo do contador
        );
        port (
            clock, zera_as, zera_s, conta: in std_logic;
            Q: out std_logic_vector (natural(ceil(log2(real(M))))-1
 downto 0);
```

```
54              fim, meio: out std_logic
55          );
56      end component;
57
58      -- REGISTRADOR
59      component registrador_n is
60          generic (
61             constant N: integer := 8
62          );
63          port (
64             clock:  in  std_logic;
65             clear:  in  std_logic;
66             enable: in  std_logic;
67             D:      in  std_logic_vector (N-1 downto 0);
68             Q:      out std_logic_vector (N-1 downto 0)
69          );
70      end component;
71
72      -- SINAIS
73      signal dist_s : std_logic_vector(11 downto 0);
74
75  begin
76
77      -- INSTANCIAS
78      U1: gerador_pulso
79              generic map(500)
80              port map(-- Entradas
81                       clock  => clock,
82                       reset  => zera,
83                       gera   => gera,
84                       para   => '0',
85                       -- Saidas
86                       pulso  => trigger,
87                       pronto => open);
88
89      U2: contadorg_m generic map(M=>4096)
90                          port map(clock => clock,
91                          zera_as => zera,
92                          zera_s => '0',
93                          conta => conta,
94                          Q => dist_s,
95                          fim => fim,
96                          meio => open );
97
98      --U2: contador_bcd_4digitos
99      --         port map(-- Entradas
100     --                  clock => clock,
101     --                 zera  => zera,
102     --                  conta => conta,
103     --                  -- Saidas
104     --                  dig3  => open,
105     --                  dig2  => dist_s(11 downto 8),
106     --
107    --                  dig1  => dist_s(7 downto 4),
108     --                  dig0  => dist_s(3 downto 0),
109     --                  fim   => fim);
110
```

```
111     U3: registrador_n
112             generic map(12)
113             port map(-- Entradas
114                      clock  => clock,
115                      clear  => zera,
116                      enable => registra,
117                      D      => dist_s,
118                      -- Saidas
119                      Q      => distancia);
120
121 end architecture;
```

13.3 Testbench

13.3.1 Security System Testbench

```
1  library ieee;
2  use ieee.std_logic_1164.all;
3  use ieee.numeric_std.all;
4
5  entity sistema_seguranca_tb is
6  end entity;
7
8  architecture tb of sistema_seguranca_tb is
9      -- Inputs
10     signal clock_in   : std_logic := '0';
11     signal reset_in   : std_logic := '0';
12     signal ligar_in   : std_logic := '0';
13     signal echo_in    : std_logic := '0';
14     signal sel_mux_in : std_logic_vector(1 downto 0) := "00";
15
16     -- Outputs
17     signal trigger_out      : std_logic := '0';
18     signal pwm_out          : std_logic := '0';
19     signal saida_serial_out : std_logic := '0';
20     signal alerta_prox_out  : std_logic := '0';
21
22     -- Controle do clock
23     signal keep_simulating : std_logic := '0';
24     constant clockPeriod   : time      := 20 ns;
25
26     -- Array de casos de teste
27   type caso_teste_type is record
28     id    : natural;
29     tempo : integer;
30   end record;
31
32   type casos_teste_array is array (natural range <>) of
       caso_teste_type;
33   constant casos_teste : casos_teste_array :=
34     (
35       (1, 100),
36       (2, 231),
37             (3, 300),
38             (4, 392),
39             (5, 492),
```

```
40              (6, 543),
41              (7, 638)
42        -- inserir aqui outros casos de teste (inserir "," na linha
      anterior)
43      );
44
45    signal larguraPulso: time := 1 ns;
46
47  begin
48
49      -- Gerador de clock
50      clock_in <= (not clock_in) and keep_simulating after
      clockPeriod/2;
51
52      -- Instancia
53      DUT: entity work.sistema_seguranca (sistema_seguranca_arch)
54          port map(
55              -- Inputs
56              clock   => clock_in,
57              reset   => reset_in,
58              ligar   => ligar_in,
59              echo    => echo_in,
60              sel_mux => sel_mux_in,
61              -- Outputs
62              trigger            => trigger_out,
63              pwm                => pwm_out,
64              saida_serial       => saida_serial_out,
65              saida_serial_ch    => open,
66              saida_serial_mqtt  => open,
67              pwm_ch             => open,
68              alerta_proximidade => alerta_prox_out,
69              alerta_prox_mqtt   => open,
70              db_transmitir      => open,
71              db_medir           => open,
72              display0           => open,
73              display1           => open,
74              display2           => open,
75              display3           => open,
76              display4           => open,
77              display5           => open
78          );
79
80      -- Estimulo
81      stim: process is
82      begin
83          assert false report "Inicio das simulacoes" severity note;
84      keep_simulating <= '1';
85
86      ---- valores iniciais ----------------
87      echo_in  <= '0';
88
89      ---- inicio: reset ----------------
90      wait for 2*clockPeriod;
91      reset_in <= '1';
92      wait for 2 us;
93      reset_in <= '0';
94      wait until falling_edge(clock_in);
```

```
95
96      ---- espera de 100us
97      wait for 100 us;
98
99          wait until falling_edge(clock_in);
100         ligar_in <= '1';
101
102     ---- loop pelos casos de teste
103     for i in casos_teste'range loop
104       -- 1) determina largura do pulso echo
105       assert false report "Caso de teste " & integer'image(
        casos_teste(i).id) & ": " &
106         integer'image(casos_teste(i).tempo) & "us" severity note;
107       larguraPulso <= casos_teste(i).tempo * 1 us; -- caso de teste
         "i"
108
109       -- 2) espera por 400us (tempo entre trigger e echo)
110       wait for 400 us;
111
112       -- 3) gera pulso de echo (largura = larguraPulso)
113       echo_in <= '1';
114       wait for larguraPulso;
115       echo_in <= '0';
116
117       -- 4) espera entre casos de tese
118       wait for 20 ms;
119
120     end loop;
121
122     ---- final dos casos de teste da simulacao
123     assert false report "Fim das simulacoes" severity note;
124     keep_simulating <= '0';
125
126     wait; -- fim da simula  o: aguarda indefinidamente (n o
        retirar esta linha)
127
128     end process;
129
130 end architecture;
```

13.4 Processing Interface

```
1  // Projeto: Sistema de Seguran a (T3BB5)
2  // Gabriel Pereira de Carvalho
3  // Ot vio Felipe de Freitas
4  // Willian Abe Fukushima
5  //-----------------------------------------------------------------
6  // Para interagir com interface, s o utilizados comandos de
       teclado (PUBLISH)
7  // l = entrada ligar
8  // r = entrada reset
9  // m = entrada mode
10 // x = entrada sel_mux[0]
11 // y = entrada sel_mux[1]
12 //-----------------------------------------------------------------
13
14 import java.io.IOException;
```

```
15 import java.time.format.DateTimeFormatter;
16 import java.time.LocalDateTime;
17 import mqtt.*;
18
19 //Conex o MQTT
20 String user   = "grupo1-bancadaB5";
21 String passwd = "L%40Bdygy1B5";      // manter %40
22 String broker = "3.141.193.238";
23 String port   = "80";
24 MQTTClient client;
25 String clientID;
26 //--------------------------------------------------------------
27
28 //Variaveis
29 int whichKey = -1;  // variavel mantem tecla acionada
30 Boolean estado_ligar = false;
31 Boolean estado_mode = false;
32 Boolean estado_reset = false;
33 Boolean estado_selmux0 = false;
34 Boolean estado_selmux1 = false;
35 Boolean estado_alerta = false;
36 Boolean estado_calibrando = false;
37 PFont myFont;
38 int largura = 960;
39 int altura = 600;
40 //--------------------------------------------------------------
41
42 //Setup: executado uma vez
43 void setup() {
44   // Tamanho do quadro
45   size(960, 600);
46   myFont = loadFont("myfont.vlw");
47   smooth();
48   // Conectar com MQTT
49   client = new MQTTClient(this);
50   clientID = new String("labead-mqtt-processing-" + random(0,100));
51   println("clientID=" + clientID);
52   client.connect("mqtt://" + user + ":" + passwd + "@" + broker + "
       :" + port, clientID, false);
53 }
54 //--------------------------------------------------------------
55
56 //Draw: executado continuamente
57 void draw() {
58     cursor(HAND);
59     textFont(myFont);
60     background(#B6BCB3);
61     // chama funcoes para desenhar painel do sistema de seguran a
62     drawCabecalho();
63     drawQuadro();
64 }
65 //--------------------------------------------------------------
66
67 void drawCabecalho() {
68     textAlign(CENTER);
69     textSize(30);
70     fill(0);
```

```
71      text("Controle os Widgets com o teclado", 10, 10, largura,
        altura);
72      textAlign(CENTER);
73      textSize(15);
74      fill(0);
75      text("r : E0 (reset)", 10, 40, largura, altura);
76      textAlign(CENTER);
77      textSize(15);
78      fill(0);
79      text("l: E1 (ligar)", 10, 55, largura, altura);
80      textAlign(CENTER);
81      textSize(15);
82      fill(0);
83      text("x,y: E2,E3 (sel_mux)", 10, 70, largura, altura);
84      textAlign(CENTER);
85      textSize(15);
86      fill(0);
87      text("m: E4 (modo 0 'em casa' ou 1 'fora de casa')", 10, 85,
        largura, altura);
88
89  }
90  void drawQuadro(){
91    stroke(#300A8B);
92    fill(#868489);
93    rect(250, 100, 500, 410);
94  //Widgets para Inputs
95    if(estado_ligar){//l de ligar
96      fill(#FF0303);
97      ellipse(350, 150, 70, 70);
98      textSize(12);
99      fill(0);
100     text("on", 350, 155);
101   }else{
102     fill(#B6BCB3);
103     ellipse(350, 150, 70, 70);
104     textSize(12);
105     fill(0);
106     text("off", 350, 155);
107   }
108   if(estado_mode){// m de mode
109     fill(#FF0303);
110     ellipse(350, 250, 70, 70);
111     textSize(11);
112     fill(0);
113     text("Fora de Casa", 350, 255);
114   }else{
115     fill(#B6BCB3);
116     ellipse(350, 250, 70, 70);
117     textSize(12);
118     fill(0);
119     text("Em Casa", 350, 255);
120   }
121   if(estado_reset){// r de reset
122     fill(#FF0303);
123     ellipse(350, 350, 70, 70);
124     textSize(12);
125     fill(0);
```

```
126     text("reset", 350, 355);
127   }else{
128     fill(#B6BCB3);
129     ellipse(350, 350, 70, 70);
130     textSize(12);
131     fill(0);
132     text("reset", 350, 355);
133   }
134   if(estado_selmux0){//x de sel_mux[0] (E2)
135     fill(#FF0303);
136     ellipse(300, 450, 70, 70);
137     textSize(12);
138     fill(0);
139     text("sel_mux[0]", 300, 455);
140   }else{
141     fill(#B6BCB3);
142     ellipse(300, 450, 70, 70);
143     textSize(12);
144     fill(0);
145     text("sel_mux[0]", 300, 455);
146   }
147   if(estado_selmux1){//y de sel_mux[1] (E3)
148     fill(#FF0303);
149     ellipse(400, 450, 70, 70);
150     textSize(12);
151     fill(0);
152     text("sel_mux[1]", 400, 455);
153   }else{
154     fill(#B6BCB3);
155     ellipse(400, 450, 70, 70);
156     textSize(12);
157     fill(0);
158     text("sel_mux[1]", 400, 455);
159   }
160   //-----------------------------------------------------------------
161   stroke(255,0,0);
162   strokeWeight(5);
163   line(500,100,500,510);
164   //Widgets para Outputs
165   if(estado_alerta){
166     stroke(#300A8B);
167     fill(#FF0303);
168     ellipse(600, 150, 70, 70);
169     textSize(12);
170     fill(0);
171     text("alerta on", 600, 155);
172   }else{
173     stroke(#300A8B);
174     fill(#B6BCB3);
175     ellipse(600, 150, 70, 70);
176     textSize(12);
177     fill(0);
178     text("alerta off", 600, 155);
179   }
180   if(estado_calibrando){
181     stroke(#300A8B);
182     fill(#FF0303);
```

```
183     ellipse(600, 250, 70, 70);
184     textSize(12);
185     fill(0);
186     text("calibrando...", 600, 255);
187   }else{
188     stroke(#300A8B);
189     if(estado_ligar){
190       fill(#00FF12);
191       ellipse(600, 250, 70, 70);
192       textSize(12);
193       fill(0);
194       text("calibrado", 600, 255);
195     }else{
196       fill(#B6BCB3);
197       ellipse(600, 250, 70, 70);
198     }
199   }
200   //---------------------------------------------------------------
201 }
202
203 //keyPressed: processa entrada por teclado
204 void keyPressed() {
205   whichKey = key;
206   if(whichKey == 114){
207     estado_reset = !estado_reset;
208     if(estado_reset){
209       client.publish(user + "/E0", "1");
210     }else{
211       client.publish(user + "/E0", "0");
212     }
213   }
214   if(whichKey == 108){
215     estado_ligar = !estado_ligar;
216     if(estado_ligar){
217       client.publish(user + "/E1", "1");
218     }else{
219       client.publish(user + "/E1", "0");
220     }
221   }
222   if(whichKey == 120){
223     estado_selmux0 = !estado_selmux0;
224     if(estado_selmux0){
225       client.publish(user + "/E2", "1");
226     }else{
227       client.publish(user + "/E2", "0");
228     }
229   }
230   if(whichKey == 121){
231     estado_selmux1 = !estado_selmux1;
232     if(estado_selmux1){
233       client.publish(user + "/E3", "1");
234     }else{
235       client.publish(user + "/E3", "0");
236     }
237   }
238   if(whichKey == 109){
239     estado_mode = !estado_mode;
```

```
240     if(estado_mode){
241       client.publish(user + "/E4", "1");
242     }else{
243       client.publish(user + "/E4", "0");
244     }
245   }
246 }
247 //----------------------------------------------------------------
248
249 //Fun   es para MQTT
250 void clientConnected() {
251   println("cliente conectado");
252   client.subscribe(user + "/S0");//alerta movimento
253   client.subscribe(user + "/S1");//calibrando
254 }
255 void messageReceived(String topic, byte[] payload) {
256   char ultimoChar = topic.charAt(topic.length()-1);
257   String dados = new String(payload);
258
259   if(ultimoChar == '0'){//mensagem veio de S0 (alerta_mov)
260     if(Integer.parseInt(dados) == 1){//alerta esta ativado
261       estado_alerta = true;
262       DateTimeFormatter dtf = DateTimeFormatter.ofPattern("yyyy/MM/
      dd HH:mm:ss");
263       LocalDateTime now = LocalDateTime.now();
264       println("ALERTA DE MOVIMENTO : " + dtf.format(now));
265     }else{
266       estado_alerta = false;
267     }
268   }
269   if(ultimoChar == '1'){//mensagem veio de S1 (calibrando)
270     if(Integer.parseInt(dados) == 1){//alerta esta ativado
271       estado_calibrando = true;
272     }else{
273       estado_calibrando = false;
274     }
275   }
276 }
277 void connectionLost() {
278   println("conexao perdida");
279 }
280 //----------------------------------------------------------------
```

Part IV

Week 4: Preparation for Project Fair

Chapter 14

Objectives

In this fourth week of the project, our goals were:

- Add password functionality to arm/disarm the circuit
- Add data persistence functionalities and sound effects to the Dashboard `Processing`

Figure 14.1: Project Logo

Chapter 15

Documentation

15.1 Control Unit

Additional states were added to the state machine to accommodate the implementation of passwords. Two new states were needed to authorize arming and disarming. Up to the planning submission, when turning off the system, the machine transitions directly to `idle`, so a password is only required if the user decides to reset the circuit after motion detection in the "away" mode.

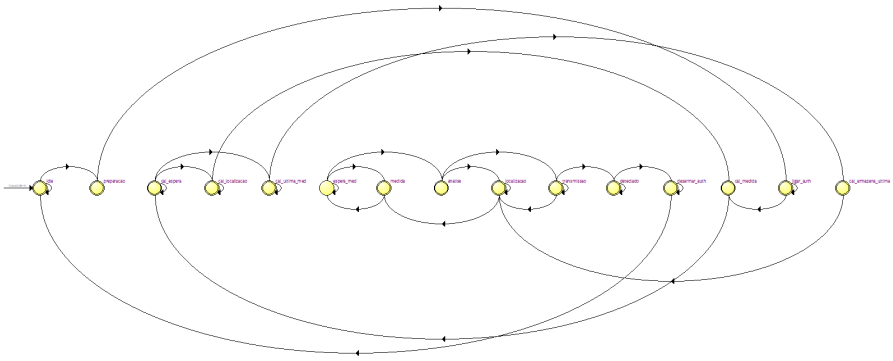

Figure 15.1: State machine with the new states `arm_auth` and `disarm_auth`.

`semana4/Images/fsm-logic.png`

Figure 15.2: New state logic to accommodate the two new added states.

15.2 Data Flow

The data flow did not change to accommodate password input compared to the previous week. However, for ease of reference, images for the data flow are also presented in this document.

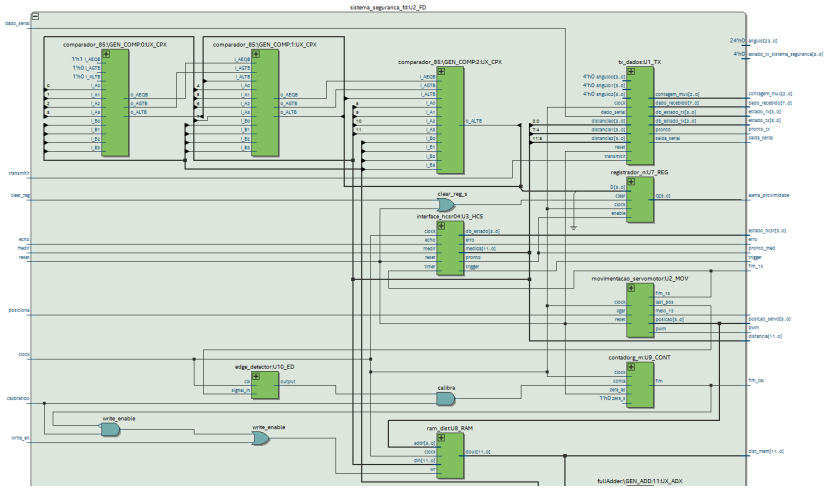

Figure 15.3: As the data flow has several components, it has been divided into two images for better visualization.

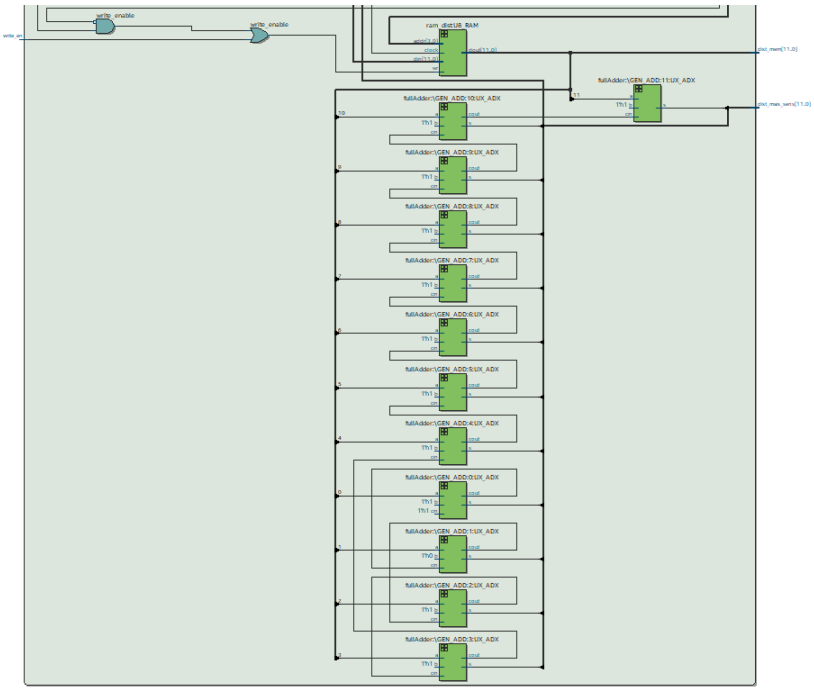

Figure 15.4: As the data flow has several components, it has been divided into two images for better visualization.

Chapter 16

Dashboard Processing

A notable change from the DashBoard Processing developed in week 3 to the version delivered this week is that we are no longer using MQTT messages to communicate with LabEAD, but rather for communication with the LabHome setup made by Otávio. We are using the same broker, but the topics used are exclusive to the LabHome setup.

The purpose of the DashBoard Processing is to serve as the main means of interaction with the project, both for testing purposes and for end users.

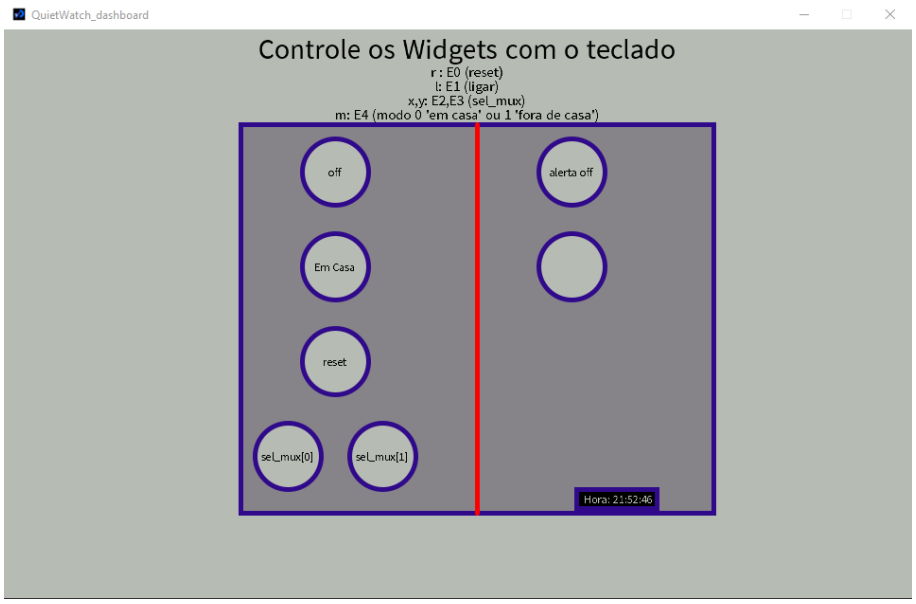

Figure 16.1: Processing dashboard

On the left side of the DashBoard, we have the input widgets.

- `reset`
- `turn on`
- `mode`
- `disarm`

These widgets are all controlled by the keyboard, typing the keys r, l, m, or d respectively to change the value of the input. It is worth noting that besides being publishers, these widgets are also subscribers to their respective topics.

Thus, the DashBoard can be used simultaneously by multiple users without one affecting another's experience. When the circuit is waiting for the password, any key typed on the keyboard will be considered a character of the password. In other words, the password can contain the characters r, l, m, or d without any issues.

The typed string is sent to the RX topic of the Arduino, where it is compared with the password.

Persistence with txt file

Firstly, the functionality of data event persistence was implemented. The Processing `PrintWriter` library allows saving data to a `txt` file located in the project folder.

```
Setup Processing 30/11/2021 horario 12:18:9
Reset  12:18:43
Recebido Reset  12:18:44
Ligar  12:18:47
Recebido Ligar  12:18:48
Inserir Senha  12:18:48
SENHA ENVIADA  1234
Calibrando  12:18:53
ALERTA DE MOVIMENTO  12:19:8
Modo Fora de Casa  12:19:19
Recebido Modo Fora de Casa  12:19:19
Modo Em Casa  12:19:29
Recebido Modo Em Casa  12:19:29
Inserir Senha  12:19:30
SENHA ENVIADA  1234
Reset  12:19:34
Recebido Reset  12:19:35
Ligar  12:19:35
Recebido Ligar  12:19:36
Inserir Senha  12:19:36
SENHA ENVIADA  1234
Calibrando  12:19:39
Modo Fora de Casa  12:19:40
Recebido Modo Fora de Casa  12:19:40
Recebido Reset  12:19:54
Inserir Senha  12:19:55
SENHA ENVIADA  1234
Calibrando  12:19:59
```

```
29 ALERTA DE MOVIMENTO  12:20:11
30 Desarmado  12:20:25
31 Recebido Desarmado  12:20:25
32 Inserir Senha  12:20:26
33 SENHA ENVIADA  1234
34 Calibrando  12:20:30
```

This file records events of changes in the `turn on`, `reset`, and `mode` inputs as well as in the `motion_alert` and `calibrating` outputs, and also requests and sends of `password`. In our project, it is worth noting that every time the Processing interface is executed, the `PrintWriter` writes a new log file. Thus, it is possible to accumulate several different log files for various different usage sessions.

It is also worth noting that events generated by other users who are publishing on the same MQTT topics are also stored in the log file. Thus, all users of the system who use the DashBoard in the same time interval will obtain equivalent logs.

Sound Effects with Minim library

Secondly, the `Minim` library was used to produce sounds with the Dashboard. Thus, the user has auditory feedback during calibration and when the system enters alarm state. Both effects (`calibrating_sound` and `alarm_sound`) were downloaded from the `ZapSplat` platform and are in the project folder in `mp3` format.

The `alarm_sound` is played only when movement is detected in the `away from home` mode. When the user is present in the residence, movement is expected, so the events are simply recorded for later analysis. However, in the `away from home` mode, the sound effect aims to alert the user of a possible intrusion.

The codes are located in the final chapter, and the project `zip` folder was submitted along with this document.

To see more details of the project's DashBoard in operation, check the Tests section of this report or the video we prepared for the Projects Fair.

Chapter 17

Tests

In this chapter, we will present an example of QuietWatch operation and the corresponding log file that persisted the event data during the test.

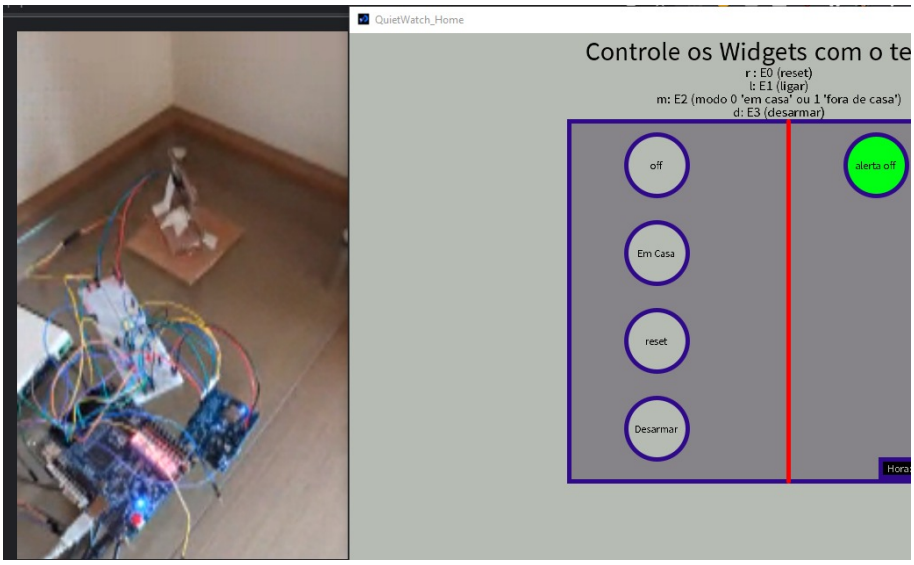

Figure 17.1: Start of the test

The first step to begin the test is to trigger the `reset` signal to ensure that the circuit is in the `idle` state before starting the test routine.

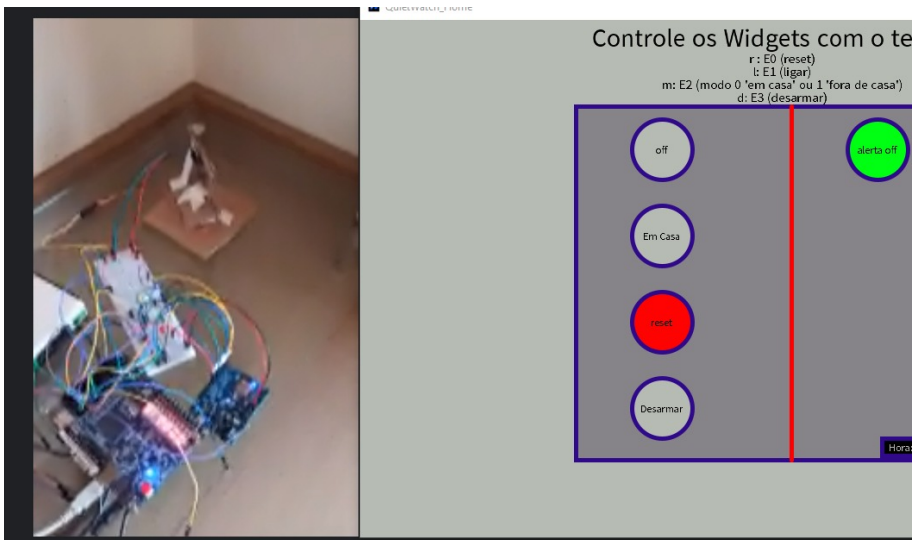

Figure 17.2: Reset activation

After triggering `reset`, we must select the operation mode of the circuit: `at home` or `away from home`, and then trigger the `turn on` input to start the normal system flow.

Note that when triggering `turn on`, the system requests the insertion of the password. Simply enter the password defined in the Processing Dashboard to proceed, in this example 1234.

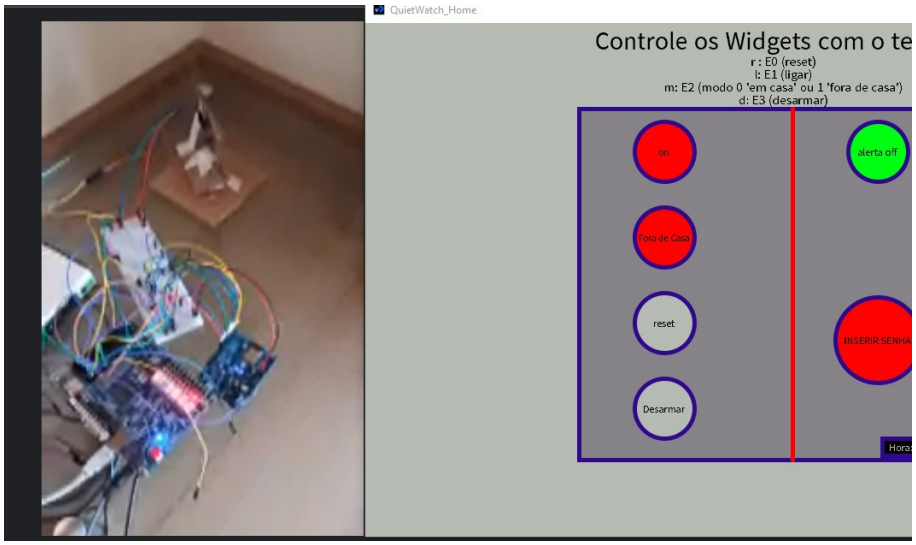

Figure 17.3: Selection of the `away from home` mode and activation of `turn on`

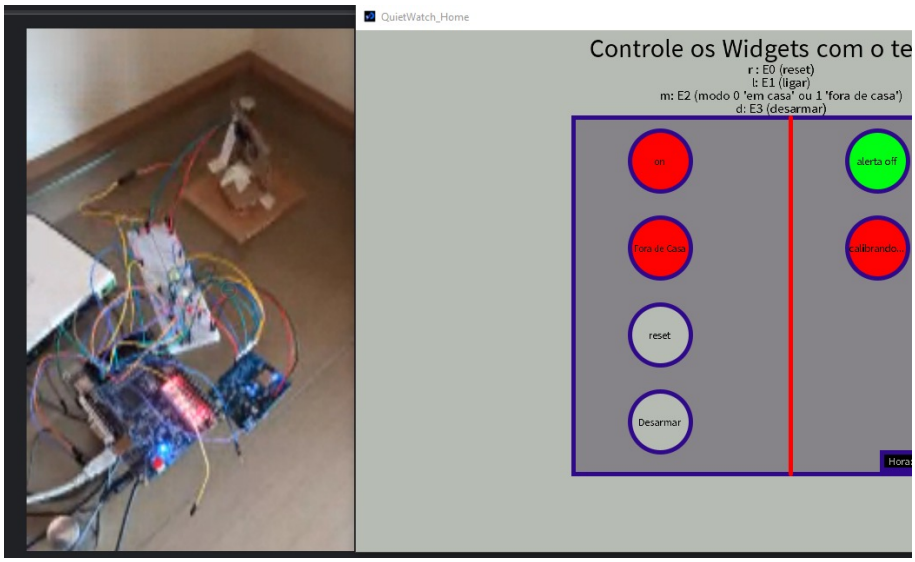

Figure 17.4: Calibration state

During the first scan, the system is being calibrated, as we can observe by the activation of the `calibrating` output in the Processing Dashboard.

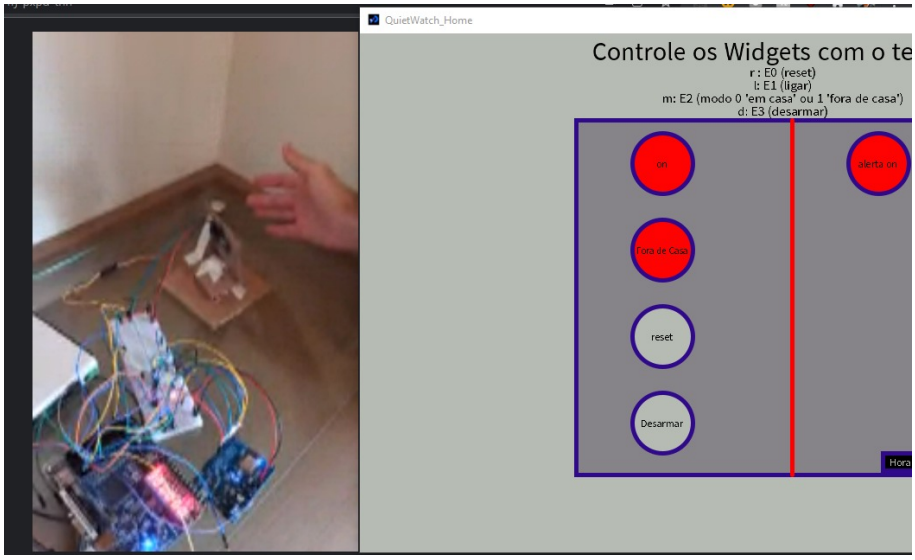

Figure 17.5: Movement detection

When inserting an object into the sensor's field of view, we observe that the `alarm_mov` output is triggered as expected, and the Processing Dashboard plays

the `alarm_sound.mp3` file.

To resume the system's operation, we trigger the `disarm` input. The system performs a new calibration, and the loop of operation is resumed.

```
Setup Processing 2/12/2021 horario 10:28:30
Reset  10:29:1
Recebido Reset  10:29:2
Modo Fora de Casa  10:29:4
Recebido Modo Fora de Casa  10:29:5
Ligar  10:29:6
Recebido Ligar  10:29:6
Inserir Senha  10:29:6
SENHA ENVIADA  1234
Calibrando  10:29:9
ALERTA DE MOVIMENTO  10:29:24
Desarmado  10:29:31
Recebido Desarmado  10:29:31
Inserir Senha  10:29:31
SENHA ENVIADA  1234
Calibrando  10:29:34
Reset  10:29:50
Recebido Reset  10:29:50
```

Chapter 18

Code

Our codes can be found in full on our Github repository.

18.1 Quartus Project

18.1.1 Complete Circuit

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity sistema_seguranca is
  port(
    -- Inputs
    clock       : in std_logic;
    reset       : in std_logic;
    ligar       : in std_logic;
    echo        : in std_logic;
    dado_serial : in std_logic;
    mode        : in std_logic;
    senha_ok    : in std_logic;
    desarmar    : in std_logic;
    sel_mux     : in std_logic_vector(1 downto 0);
    -- Outputs
    trigger            : out std_logic;
    db_trigger         : out std_logic;
    db_echo            : out std_logic;
    pwm                : out std_logic;
    saida_serial       : out std_logic;
    saida_serial_ch    : out std_logic;
    saida_serial_mqtt  : out std_logic;
    pwm_ch             : out std_logic;
    alerta_proximidade : out std_logic;
    alerta_prox_mqtt   : out std_logic;
    db_transmitir      : out std_logic;
    db_medir           : out std_logic;
    calibrando         : out std_logic;
    alerta_mov         : out std_logic;
```

```
32      inserir_senha      : out std_logic;
33      db_mode            : out std_logic;
34      db_fim_2s          : out std_logic;
35      db_erro            : out std_logic;
36      display0           : out std_logic_vector(6 downto 0);
37      display1           : out std_logic_vector(6 downto 0);
38      display2           : out std_logic_vector(6 downto 0);
39      display3           : out std_logic_vector(6 downto 0);
40      display4           : out std_logic_vector(6 downto 0);
41      display5           : out std_logic_vector(6 downto 0)
42    );
43  end entity;
44
45  architecture sistema_seguranca_arch of sistema_seguranca is
46
47    -- Unidade de Controle
48    component sistema_seguranca_uc is
49      port (
50        -- Inputs
51        clock      : in std_logic;
52        reset      : in std_logic;
53        ligar      : in std_logic;
54        fim_1s     : in std_logic;
55        pronto_med : in std_logic;
56        pronto_tx  : in std_logic;
57        fim_cal    : in std_logic;
58        alerta     : in std_logic;
59        mode       : in std_logic;
60        auth       : in std_logic;
61        desarmar   : in std_logic;
62        erro       : in std_logic;
63        -- Outputs
64        zera       : out std_logic;
65        posiciona  : out std_logic;
66        medir      : out std_logic;
67        transmitir : out std_logic;
68        calibrando : out std_logic;
69        write_en   : out std_logic;
70        alerta_out : out std_logic;
71        clear_reg  : out std_logic;
72        db_estado  : out std_logic_vector(3 downto 0)
73      );
74    end component;
75
76    -- Fluxo de Dados
77    component sistema_seguranca_fd is
78      port (
79        -- Inputs
80        clock       : in std_logic;
81        reset       : in std_logic;
82        zera        : in std_logic;
83        ligar       : in std_logic;
84        medir       : in std_logic;
85        posiciona   : in std_logic;
86        transmitir  : in std_logic;
87        echo        : in std_logic;
88        dado_serial : in std_logic;
```

```
89      calibrando   : in std_logic;
90      write_en     : in std_logic;
91      clear_reg    : in std_logic;
92      clear_sig    : in std_logic;
93      mode         : in std_logic;
94      auth         : in std_logic;
95      -- Outputs
96      pwm                          : out std_logic;
97      trigger                      : out std_logic;
98      saida_serial                 : out std_logic;
99      pronto_tx                    : out std_logic;
100     alerta_proximidade           : out std_logic;
101     fim_1s                       : out std_logic;
102     meio_1s                      : out std_logic;
103     pronto_med                   : out std_logic;
104     fim_cal                      : out std_logic;
105     erro                         : out std_logic;
106     ligar_reg                    : out std_logic;
107     mode_reg                     : out std_logic;
108     contagem_mux                 : out std_logic_vector(2 downto
    0);
109     estado_hcsr                  : out std_logic_vector(3 downto
    0);
110     estado_tx_sistema_seguranca : out std_logic_vector(3 downto
    0);
111     estado_rx                    : out std_logic_vector(3 downto
    0);
112     estado_tx                    : out std_logic_vector(3 downto
    0);
113     posicao_servo                : out std_logic_vector(3 downto
    0);
114     dado_recebido                : out std_logic_vector(7 downto
    0);
115     distancia                    : out std_logic_vector(11 downto
    0);
116     dist_mem                     : out std_logic_vector(11 downto
    0);
117     dist_mais_sens               : out std_logic_vector(11 downto
    0);
118     angulo                       : out std_logic_vector(23 downto
    0)
119   );
120 end component;
121
122 -- Multiplexador 4x1
123 component mux_4x1_n is
124   generic (
125     constant BITS: integer := 4
126   );
127   port (
128     D0 :      in  std_logic_vector (BITS-1 downto 0);
129     D1 :      in  std_logic_vector (BITS-1 downto 0);
130     D2 :      in  std_logic_vector (BITS-1 downto 0);
131     D3 :      in  std_logic_vector (BITS-1 downto 0);
132     SEL:      in  std_logic_vector (1 downto 0);
133     MUX_OUT: out std_logic_vector (BITS-1 downto 0)
134   );
```

```
135   end component;
136
137   -- Display de 7 segmentos
138   component hex7seg is
139     port (
140       hexa : in  std_logic_vector(3 downto 0);
141       sseg : out std_logic_vector(6 downto 0)
142     );
143   end component;
144
145   -- Unidade de gerenciamento de senha
146   component pswd_handler is
147     port (
148       -- Input
149       clock    : in std_logic;
150       reset    : in std_logic;
151       ligar    : in std_logic;
152       mode     : in std_logic;
153       desarmar : in std_logic;
154       pswd_ok  : in std_logic;
155       -- Output
156       wait_pswd : out std_logic;
157       auth      : out std_logic;
158       -- Debug
159       db_event : out std_logic;
160       db_state : out std_logic_vector(3 downto 0)
161     );
162   end component;
163
164   -- Sinais
165   signal pronto_tx_s                              : std_logic;
166   signal fim_1s_s, zera_s, reset_fd               : std_logic;
167   signal meio_1s_s, pede_senha_s                  : std_logic;
168   signal posiciona_s, medir_s                     : std_logic;
169   signal saida_serial_s, pwm_s                    : std_logic;
170   signal alerta_proximidade_s                     : std_logic;
171   signal transmitir_s, pronto_med_s               : std_logic;
172   signal fim_cal_s, calibrando_s, write_en_s      : std_logic;
173   signal trigger_s, mode_s, alerta_s              : std_logic;
174   signal erro_s, clear_reg_s, ligar_reg_s         : std_logic;
175   signal mode_reg_s                               : std_logic;
176   signal contagem_mux_3bits                       :
        std_logic_vector(2 downto 0);
177   signal contagem_mux_4bits                       :
        std_logic_vector(3 downto 0);
178   signal sistema_seguranca_estado, posicao_4bits  :
        std_logic_vector(3 downto 0);
179   signal estado_hcsr, estado_tx_sistema_seguranca :
        std_logic_vector(3 downto 0);
180   signal estado_rx, estado_tx                     :
        std_logic_vector(3 downto 0);
181   signal dado_recebido_s                          :
        std_logic_vector(7 downto 0);
182   signal distancia_bcd                            :
        std_logic_vector(11 downto 0);
183   signal dist_mem                                 :
        std_logic_vector(11 downto 0);
```

```
184   signal dist_sens                                    :
        std_logic_vector(11 downto 0);
185   signal angulo_bcd_hex                               :
        std_logic_vector(23 downto 0);
186
187   -- Sinais unidade de controle
188   signal reset_auth : std_logic;
189
190   -- Sinais gerencimento de senha
191   signal event_ph : std_logic;
192   signal auth_s   : std_logic;
193   signal ph_state : std_logic_vector(3 downto 0);
194
195   -- Saidas dos multiplexadores
196   signal m0_out, m1_out, m2_out, m3_out, m4_out, m5_out :
        std_logic_vector(3 downto 0);
197
198 begin
199
200   -- Logica de sinais
201   reset_auth         <= reset and auth_s;
202   reset_fd           <= reset or zera_s;
203   mode_s             <= mode;
204   contagem_mux_4bits <= '0' & contagem_mux_3bits;
205
206   -- Instancias
207   U1_UC: sistema_seguranca_uc
208     port map(
209       -- Inputs
210       clock      => clock,
211       reset      => reset_auth,
212       ligar      => ligar_reg_s,
213       fim_1s     => fim_1s_s,
214       pronto_med => pronto_med_s,
215       pronto_tx  => pronto_tx_s,
216       fim_cal    => fim_cal_s,
217       alerta     => alerta_proximidade_s,
218       mode       => mode_reg_s,
219       auth       => auth_s,
220       desarmar   => desarmar,
221       erro       => erro_s,
222       -- Outputs
223       zera       => zera_s,
224       posiciona  => posiciona_s,
225       medir      => medir_s,
226       transmitir => transmitir_s,
227       calibrando => calibrando_s,
228       write_en   => write_en_s,
229       alerta_out => alerta_s,
230       clear_reg  => clear_reg_s,
231       db_estado  => sistema_seguranca_estado
232     );
233
234   U2_FD: sistema_seguranca_fd
235     port map(
236       -- Inputs
237       clock       => clock,
```

```
238       reset       => reset_fd,
239       zera        => zera_s,
240       ligar       => ligar,
241       medir       => medir_s,
242       posiciona   => posiciona_s,
243       transmitir  => transmitir_s,
244       echo        => echo,
245       dado_serial => dado_serial,
246       calibrando  => calibrando_s,
247       write_en    => write_en_s,
248       clear_reg   => clear_reg_s,
249       clear_sig   => reset_auth,
250       mode        => mode_s,
251       auth        => auth_s,
252       -- Outputs
253       pwm                          => pwm_s,
254       trigger                      => trigger_s,
255       saida_serial                 => saida_serial_s,
256       pronto_tx                    => pronto_tx_s,
257       alerta_proximidade           => alerta_proximidade_s,
258       fim_1s                       => fim_1s_s,
259       meio_1s                      => meio_1s_s,
260       pronto_med                   => pronto_med_s,
261       fim_cal                      => fim_cal_s,
262       erro                         => erro_s,
263       ligar_reg                    => ligar_reg_s,
264       mode_reg                     => mode_reg_s,
265       contagem_mux                 => contagem_mux_3bits,
266       estado_hcsr                  => estado_hcsr,
267       estado_tx_sistema_seguranca => estado_tx_sistema_seguranca,
268       estado_rx                    => estado_rx,
269       estado_tx                    => estado_tx,
270       posicao_servo                => posicao_4bits,
271       dado_recebido                => dado_recebido_s,
272       distancia                    => distancia_bcd,
273       dist_mem                     => dist_mem,
274       dist_mais_sens               => dist_sens,
275       angulo                       => angulo_bcd_hex
276     );
277
278   U3_PSWD: pswd_handler
279     port map (
280       -- Input
281       clock    => clock,
282       reset    => reset,
283       ligar    => ligar,
284       mode     => mode,
285       desarmar => desarmar,
286       pswd_ok  => senha_ok,
287       -- Output
288       wait_pswd => inserir_senha,
289       auth      => auth_s,
290       -- Debug
291       db_event => event_ph,
292       db_state => ph_state
293     );
294
```

```
295   M0: mux_4x1_n
296     generic map(4)
297     port map(
298       -- Inputs
299       D0  => distancia_bcd(3 downto 0), --SEL_MUX=00 (distancia0)
300       D1  => dado_recebido_s(3 downto 0), --SEL_MUX=01 (DADO_RX1)
301       D2  => dist_mem(3 downto 0), --SEL_MUX=10 (dist_mem0)
302       D3  => angulo_bcd_hex(3 downto 0),--SEL_MUX=11 (angulo0)
303       SEL => sel_mux,
304       -- Output
305       MUX_OUT => m0_out
306     );
307
308   M1: mux_4x1_n
309     generic map(4)
310     port map(
311       -- Inputs
312       D0  => distancia_bcd(7 downto 4),--SEL_MUX=00 (distancia1)
313       D1  => dado_recebido_s(7 downto 4), --SEL_MUX=01 (dado_RX1)
314       D2  => dist_mem(7 downto 4), --SEL_MUX=10 (dist_mem1)
315       D3  => angulo_bcd_hex(11 downto 8),--SEL_MUX=11 (angulo1)
316       SEL => sel_mux,
317       -- Output
318       MUX_OUT => m1_out
319     );
320
321   M2: mux_4x1_n
322     generic map(4)
323     port map(
324       -- Inputs
325       D0  => distancia_bcd(11 downto 8),--SEL_MUX=00 (distancia2)
326       D1  => estado_rx, --SEL_MUX=01 (estado_rx)
327       D2  => dist_mem(11 downto 8), -- --SEL_MUX=10 (dist_mem2)
328       D3  => angulo_bcd_hex(19 downto 16),--SEL_MUX=11 (angulo2)
329       SEL => sel_mux,
330       -- Output
331       MUX_OUT => m2_out
332     );
333
334   M3: mux_4x1_n
335     generic map(4)
336     port map(
337       -- Inputs
338       D0  => "0000",--SEL_MUX=00 a definir...
339       D1  => dist_sens(3 downto 0), --SEL_MUX=00 (DADO_TX0)
340       D2  => "0000", --SEL_MUX=00 (dado_tx1)
341       D3  => "0000",--SEL_MUX=11 a definir...
342       SEL => sel_mux,
343       -- Output
344       MUX_OUT => m3_out
345     );
346
347   M4: mux_4x1_n
348     generic map(4)
349     port map(
350       -- Inputs
351       D0  => estado_hcsr,
```

```
352       D1  => dist_sens(7 downto 4), -- DADO_TX1
353       D2  => contagem_mux_4bits,
354       D3  => ph_state,
355       SEL => sel_mux,
356       -- Output
357       MUX_OUT => m4_out
358     );
359
360   M5: mux_4x1_n
361     generic map(4)
362     port map(
363       -- Inputs
364       D0  => X"0",
365       D1  => dist_sens(11 downto 8),
366       D2  => estado_tx_sistema_seguranca,
367       D3  => sistema_seguranca_estado,
368       SEL => sel_mux,
369       -- Output
370       MUX_OUT => m5_out
371     );
372
373   H0: hex7seg port map(hexa => m0_out, sseg => display0);
374   H1: hex7seg port map(hexa => m1_out, sseg => display1);
375   H2: hex7seg port map(hexa => m2_out, sseg => display2);
376   H3: hex7seg port map(hexa => m3_out, sseg => display3);
377   H4: hex7seg port map(hexa => m4_out, sseg => display4);
378   H5: hex7seg port map(hexa => m5_out, sseg => display5);
379
380   -- Outputs
381   pwm    <= pwm_s;
382   pwm_ch <= pwm_s;
383
384   trigger <= trigger_s;
385
386   calibrando <= calibrando_s;
387
388   alerta_mov <= alerta_s or (alerta_proximidade_s and (not
        calibrando_s));
389
390   saida_serial      <= saida_serial_s;
391   saida_serial_ch   <= saida_serial_s;
392   saida_serial_mqtt <= saida_serial_s;
393
394   alerta_proximidade <= alerta_proximidade_s;
395   alerta_prox_mqtt   <= alerta_proximidade_s;
396
397   db_medir   <= medir_s;
398   db_trigger <= trigger_s;
399   db_echo    <= echo;
400   db_mode    <= mode_s;
401   db_fim_2s  <= fim_1s_s;
402   db_erro    <= event_ph;
403
404 end architecture;
```

18.1.2 Control Unit

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity sistema_seguranca_uc is
    port (
        -- Inputs
        clock      : in std_logic;
        reset      : in std_logic;
        ligar      : in std_logic;
        fim_1s     : in std_logic;
        pronto_med : in std_logic;
        pronto_tx  : in std_logic;
        fim_cal    : in std_logic;
        alerta     : in std_logic;
        mode       : in std_logic;
        auth       : in std_logic;
        desarmar   : in std_logic;
        erro       : in std_logic;
        -- Outputs
        zera       : out std_logic;
        posiciona  : out std_logic;
        medir      : out std_logic;
        transmitir : out std_logic;
        calibrando : out std_logic;
        write_en   : out std_logic;
        alerta_out : out std_logic;
        clear_reg  : out std_logic;
        db_estado  : out std_logic_vector(3 downto 0)
    );
end entity;

architecture sistema_seguranca_uc_arch of sistema_seguranca_uc is

    type tipo_estado is
        (
            idle, preparacao, cal_medida,
            cal_espera, cal_localizacao, cal_ultima_med,
            cal_armazena_ultima, medida, espera_med,
            transmissao, localizacao, analise, detectado,
            espera_auth
        );

    signal Eatual : tipo_estado;
    signal Eprox  : tipo_estado;

begin

    -- Memoria de estado
    process(clock, reset, ligar, erro)
    begin
        if reset = '1' then
            Eatual <= idle;
        elsif ligar = '0' then
            Eatual <= idle;
        elsif auth = '1' then
            Eatual <= preparacao;
```

```
58          elsif clock'event and clock = '1' then
59              Eatual <= Eprox;
60          end if;
61      end process;
62
63      -- Logica de proximo estado
64      process(ligar, fim_1s, pronto_med, pronto_tx, fim_cal, alerta,
        mode, auth, desarmar, Eatual)
65      begin
66          case Eatual is
67              when idle =>
68                  if ligar = '1' then
69                      Eprox <= preparacao;
70                  else
71                      Eprox <= idle;
72                  end if;
73
74              when preparacao => Eprox <= cal_medida;
75
76              -- Calibracao
77              when cal_medida => Eprox <= cal_espera;
78
79              when cal_espera =>
80                  if fim_cal = '1' then
81                      Eprox <= cal_ultima_med;
82                  elsif fim_1s = '1' then
83                      Eprox <= cal_localizacao;
84                  elsif pronto_med = '1' then
85                      Eprox <= cal_localizacao;
86                  else
87                      Eprox <= cal_espera;
88                  end if;
89
90              when cal_localizacao =>
91                  if fim_1s = '1' then
92                      Eprox <= cal_medida;
93                  else
94                      Eprox <= cal_localizacao;
95                  end if;
96
97              when cal_ultima_med =>
98                  if pronto_med = '1' then
99                      Eprox <= cal_armazena_ultima;
100                 elsif fim_1s = '1' then
101                     Eprox <= cal_armazena_ultima;
102                 else
103                     Eprox <= cal_ultima_med;
104                 end if;
105
106             when cal_armazena_ultima => Eprox <= localizacao;
107
108             -- Deteccao de movimento
109             when medida => Eprox <= espera_med;
110
111             when espera_med =>
112                 if fim_1s = '1' then
113                     Eprox <= localizacao;
```

```
114                 elsif pronto_med = '1' then
115                     Eprox <= analise;
116                 else
117                     Eprox <= espera_med;
118                 end if;
119 
120             when analise =>
121                 if alerta = '1' then
122                     Eprox <= transmissao;
123                 else
124                     Eprox <= localizacao;
125                 end if;
126 
127             when transmissao =>
128                 if pronto_tx = '1' and mode = '1' then
129                     Eprox <= detectado;
130                 elsif pronto_tx = '1' and mode = '0' then
131                     Eprox <= localizacao;
132                 else
133                     Eprox <= transmissao;
134                 end if;
135 
136             when detectado =>
137                 if desarmar = '1' then
138                     Eprox <= espera_auth;
139                 else
140                     Eprox <= detectado;
141                 end if;
142 
143             when localizacao =>
144                 if fim_1s = '1' then
145                     Eprox <= medida;
146                 else
147                     Eprox <= localizacao;
148                 end if;
149 
150             -- Autenticacao
151             when espera_auth =>
152                 if auth = '1' then
153                     Eprox <= preparacao;
154                 else
155                     Eprox <= espera_auth;
156                 end if;
157 
158             when others => Eprox <= idle;
159         end case;
160     end process;
161 
162     -- Logica de saida
163     with Eatual select
164         zera <= '1' when preparacao, '0' when others;
165 
166     with Eatual select
167         medir <=
168             '1' when medida,
169             '1' when cal_medida,
170             '0' when others;
```

```
171
172     with Eatual select
173         transmitir <= '1' when transmissao, '0' when others;
174
175     with Eatual select
176         posiciona <=
177             '0' when idle,
178             '0' when espera_auth,
179             '0' when detectado,
180             '1' when others;
181
182     with Eatual select
183         calibrando <=
184             '1' when cal_medida,
185             '1' when cal_espera,
186             '1' when cal_localizacao,
187             '1' when cal_ultima_med,
188             '1' when cal_armazena_ultima,
189             '0' when others;
190
191     with Eatual select
192         write_en <= '1' when cal_armazena_ultima, '0' when others;
193
194     with Eatual select
195         alerta_out <= '1' when detectado, '0' when others;
196
197     with Eatual select
198         clear_reg <= '1' when localizacao, '0' when others;
199
200     -- Debug
201     with Eatual select
202         db_estado <=
203             "0000" when idle,
204             "0001" when preparacao,
205             "0010" when medida,
206             "0011" when espera_med,
207             "0100" when transmissao,
208             "0101" when localizacao,
209             "0110" when cal_medida,
210             "0111" when cal_espera,
211             "1000" when cal_localizacao,
212             "1001" when cal_ultima_med,
213             "1010" when cal_armazena_ultima,
214             "1011" when analise,
215             "1101" when detectado,
216             "1110" when espera_auth;
217
218 end architecture;
```

18.1.3 Data Flow

```
1 library ieee;
2 use ieee.std_logic_1164.all;
3 use ieee.numeric_std.all;
4 use ieee.math_real.all;
5
6 entity sistema_seguranca_fd is
```

```
7     port (
8     -- Inputs
9     clock        : in std_logic;
10    reset        : in std_logic;
11    zera         : in std_logic;
12    ligar        : in std_logic;
13    medir        : in std_logic;
14    posiciona    : in std_logic;
15    transmitir   : in std_logic;
16    echo         : in std_logic;
17    dado_serial  : in std_logic;
18    calibrando   : in std_logic;
19    write_en     : in std_logic;
20    clear_reg    : in std_logic;
21    clear_sig    : in std_logic;
22    mode         : in std_logic;
23    auth         : in std_logic;
24    -- Outputs
25    pwm                          : out std_logic;
26    trigger                      : out std_logic;
27    saida_serial                 : out std_logic;
28    pronto_tx                    : out std_logic;
29    alerta_proximidade           : out std_logic;
30    fim_1s                       : out std_logic;
31    meio_1s                      : out std_logic;
32    pronto_med                   : out std_logic;
33    fim_cal                      : out std_logic;
34    erro                         : out std_logic;
35    ligar_reg                    : out std_logic;
36    mode_reg                     : out std_logic;
37    contagem_mux                 : out std_logic_vector(2 downto 0);
38    estado_hcsr                  : out std_logic_vector(3 downto 0);
39    estado_tx_sistema_seguranca : out std_logic_vector(3 downto 0);
40    estado_rx                    : out std_logic_vector(3 downto 0);
41    estado_tx                    : out std_logic_vector(3 downto 0);
42    posicao_servo                : out std_logic_vector(3 downto 0);
43    dado_recebido                : out std_logic_vector(7 downto 0);
44    distancia                    : out std_logic_vector(11 downto 0)
      ;
45    dist_mem                     : out std_logic_vector(11 downto 0)
      ;
46    dist_mais_sens               : out std_logic_vector(11 downto 0)
      ;
47    angulo                       : out std_logic_vector(23 downto 0)
48    );
49 end entity;
50
51 architecture sistema_seguranca_fd_arch of sistema_seguranca_fd is
52
53   -- Controle da movimenta  o do servo motor
54   component movimentacao_servomotor is
55     port (
56       -- Inputs
57       clock : in std_logic;
58       reset : in std_logic;
59       ligar : in std_logic;
60       -- Outputs
```

```
61      pwm      : out std_logic;
62      fim_1s   : out std_logic;
63      meio_1s  : out std_logic;
64      last_pos : out std_logic;
65      posicao  : out std_logic_vector(3 downto 0)
66    );
67  end component;
68
69  -- Transmissao Dados sistema_seguranca
70    component tx_dados is
71        port (
72      -- Inputs
73      clock      : in std_logic;
74      reset      : in std_logic;
75      transmitir : in std_logic;
76      dado_serial: in std_logic;
77      angulo2    : in std_logic_vector(3 downto 0);
78      angulo1    : in std_logic_vector(3 downto 0);
79      angulo0    : in std_logic_vector(3 downto 0);
80      distancia2 : in std_logic_vector(3 downto 0);
81      distancia1 : in std_logic_vector(3 downto 0);
82      distancia0 : in std_logic_vector(3 downto 0);
83      -- Outputs
84      saida_serial : out std_logic;
85      pronto       : out std_logic;
86      pronto_rx    : out std_logic;
87      contagem_mux : out std_logic_vector(2 downto 0);
88      dado_recebido: out std_logic_vector(7 downto 0);
89      -- Debug
90      db_transmitir   : out std_logic;
91      db_saida_serial : out std_logic;
92      db_estado_tx    : out std_logic_vector(3 downto 0);
93      db_estado_rx    : out std_logic_vector(3 downto 0)
94    );
95    end component;
96
97    -- Detetor de Borda
98    component edge_detector is
99        port (
100           -- Inputs
101           clk       : in   std_logic;
102           signal_in : in   std_logic;
103           -- Output
104           output : out  std_logic
105       );
106   end component;
107
108 -- Interface HCSR04
109 component interface_hcsr04
110   port (
111     -- Inputs
112     clock : in std_logic;
113     reset : in std_logic;
114     medir : in std_logic;
115     echo  : in std_logic;
116     timer : in std_logic;
117     -- Outputs
```

```
118       trigger : out std_logic;
119       pronto  : out std_logic;
120       erro    : out std_logic;
121       medida  : out std_logic_vector(11 downto 0); -- 3 digitos BCD
122       -- Debug
123       db_estado : out std_logic_vector(3 downto 0)
124     );
125   end component;
126
127   component comparador_85 is
128     port (
129       i_A3   : in  std_logic; -- i_A3, i_A2, i_A1 e i_A0
130                   -- sao os bits que formam a palavra A
131       i_B3   : in  std_logic; -- i_B3, i_B2, i_B1 e i_B0
132                   -- sao os bits que formam a palavra B
133       i_A2   : in  std_logic;
134       i_B2   : in  std_logic;
135       i_A1   : in  std_logic;
136       i_B1   : in  std_logic;
137       i_A0   : in  std_logic;
138       i_B0   : in  std_logic;
139       i_AGTB : in  std_logic; -- i_AGTB, i_ALTB e i_AEQB sao inputs
140                   -- de cascateamento
141       i_ALTB : in  std_logic;
142       i_AEQB : in  std_logic;
143       o_AGTB : out std_logic; -- Saida em alto se A>B
144       o_ALTB : out std_logic; -- Saida em alto se A<B
145       o_AEQB : out std_logic  -- Saida em alto se A=B
146     );
147   end component;
148
149   -- RAM com distancias
150   component ram_dist is
151     port (
152       -- Inputs
153       clock : in std_logic;
154       wr    : in std_logic;
155       addr  : in std_logic_vector(3 downto 0);
156       din   : in std_logic_vector(11 downto 0);
157       -- Output
158       dout : out std_logic_vector(11 downto 0)
159     );
160   end component;
161
162   -- Registrador generico
163   component registrador_n is
164     generic (
165       constant N: integer := 8
166     );
167     port (
168       clock:  in  std_logic;
169       clear:  in  std_logic;
170       enable: in  std_logic;
171       D:      in  std_logic_vector (N-1 downto 0);
172       Q:      out std_logic_vector (N-1 downto 0)
173     );
174   end component;
```

```
175
176   -- Contador Generico
177     component contadorg_m is
178         generic (
179             constant M: integer := 50 -- modulo do contador
180         );
181         port (
182             -- Inputs
183             clock   : in std_logic;
184             zera_as : in std_logic;
185             zera_s  : in std_logic;
186             conta   : in std_logic;
187             -- Outputs
188             Q    : out std_logic_vector (natural(ceil(log2(real(M))
      ))-1 downto 0);
189             fim  : out std_logic;
190             meio : out std_logic
191         );
192     end component;
193
194   -- Full Adder
195   component fullAdder is
196     port(
197       -- Inputs
198       a, b, cin : in std_logic;
199       -- Outputs
200       s, cout   : out std_logic
201     );
202   end component;
203
204   -- Sinais
205   signal clear_reg_s      : std_logic;
206   signal pronto_med_s     : std_logic;
207   signal trigger_s        : std_logic;
208   signal write_stop       : std_logic;
209   signal write_enable     : std_logic;
210   signal last_pos         : std_logic;
211   signal last_pos_ed      : std_logic;
212   signal calibra          : std_logic;
213   signal ligar_servo      : std_logic;
214   signal fim_1s_s         : std_logic;
215   signal rst_hcsr         : std_logic;
216   signal clear_ligar      : std_logic;
217   signal reset_hcsr       : std_logic;
218   signal ligar_vec        : std_logic_vector(0 downto 0);
219   signal ligar_reg_s      : std_logic_vector(0 downto 0);
220   signal mode_vec         : std_logic_vector(0 downto 0);
221   signal mode_reg_s       : std_logic_vector(0 downto 0);
222   signal agtb_vector      : std_logic_vector(3 downto 0);
223   signal altb_vector      : std_logic_vector(3 downto 0);
224   signal aeqb_vector      : std_logic_vector(3 downto 0);
225   signal altb_vector_reg  : std_logic_vector(3 downto 0);
226   signal posicao_s        : std_logic_vector(3 downto 0);
227   signal dado_recebido_s  : std_logic_vector(7 downto 0);
228   signal distancia_hcsr   : std_logic_vector(11 downto 0);
229   signal distancia_ram    : std_logic_vector(11 downto 0);
230   signal distancia_reg    : std_logic_vector(11 downto 0);
```

```
231   signal sensibilidade    : std_logic_vector(11 downto 0);
232   signal dist_sens        : std_logic_vector(11 downto 0);
233   signal carry            : std_logic_vector(12 downto 0);
234   signal angulo_rom       : std_logic_vector(23 downto 0);
235
236 begin
237
238   -- Logica de sinais
239   calibra      <= calibrando and last_pos_ed;
240   write_enable <= (calibrando and (not write_stop)) or write_en;
241   ligar_servo  <= posiciona;
242   reset_hcsr   <= reset or clear_reg or auth;
243
244   -- Inicializacao
245   agtb_vector(0) <= '0';
246   altb_vector(0) <= '0';
247   aeqb_vector(0) <= '1';
248
249   carry(0) <= '1';
250
251   sensibilidade <= not B"0000_0000_0010";
252
253   ligar_vec(0) <= ligar;
254   mode_vec(0)  <= mode;
255
256   -- Instancias
257     U1_TX: tx_dados
258         port map(
259             -- Inputs
260             clock       => clock,
261             reset       => reset,
262             transmitir  => transmitir,
263       dado_serial => dado_serial,
264             angulo2     => angulo_rom(19 downto 16),
265             angulo1     => angulo_rom(11 downto 8),
266             angulo0     => angulo_rom(3 downto 0),
267             distancia2  => distancia_hcsr(11 downto 8),
268             distancia1  => distancia_hcsr(7 downto 4),
269             distancia0  => distancia_hcsr(3 downto 0),
270             -- Outputs
271             saida_serial  => saida_serial,
272             pronto        => pronto_tx,
273       pronto_rx     => open,
274       contagem_mux  => contagem_mux,
275       dado_recebido => dado_recebido_s,
276             -- Debug
277             db_transmitir   => open,
278             db_saida_serial => open,
279             db_estado_tx    => estado_tx,
280       db_estado_rx    => estado_rx
281         );
282
283   U2_MOV: movimentacao_servomotor
284     port map(
285       -- Inputs
286       clock => clock,
287       reset => reset,
```

```
288       ligar => ligar_servo,
289       -- Outputs
290       pwm      => pwm,
291       fim_1s   => fim_1s_s,
292       meio_1s  => meio_1s,
293       last_pos => last_pos,
294       posicao  => posicao_s
295     );
296
297   U3_HCS: interface_hcsr04
298     port map(
299       -- Entradas
300       clock => clock,
301       reset => reset_hcsr,
302       medir => medir,
303       echo  => echo,
304       timer => fim_1s_s,
305       -- Saidas
306       trigger => trigger,
307       pronto  => pronto_med_s,
308       erro    => erro,
309       medida  => distancia_hcsr, -- 3 digitos BCD
310       -- Debug
311       db_estado => estado_hcsr
312     );
313
314   GEN_COMP: for i in 0 to 2 generate
315     UX_CPX: comparador_85
316       port map(
317         -- Inputs
318         i_A3 => distancia_hcsr(4*i + 3),
319         i_B3 => dist_sens(4*i + 3),
320         i_A2 => distancia_hcsr(4*i + 2),
321         i_B2 => dist_sens(4*i + 2),
322         i_A1 => distancia_hcsr(4*i + 1),
323         i_B1 => dist_sens(4*i + 1),
324         i_A0 => distancia_hcsr(4*i),
325         i_B0 => dist_sens(4*i),
326         -- Cascateamento
327         i_AGTB => agtb_vector(i),
328         i_ALTB => altb_vector(i),
329         i_AEQB => aeqb_vector(i),
330         -- Outputs
331         o_AGTB => agtb_vector(i + 1),
332         o_ALTB => altb_vector(i + 1),
333         o_AEQB => aeqb_vector(i + 1)
334       );
335   end generate;
336
337   GEN_ADD: for i in 0 to 11 generate
338     UX_ADX: fullAdder
339       port map(
340         -- Inputs
341         a   => distancia_ram(i),
342         b   => sensibilidade(i),
343         cin => carry(i),
344         -- Outputs
```

```
345         s    => dist_sens(i),
346         cout => carry(i + 1)
347       );
348   end generate;
349
350   U7_REG: registrador_n
351     generic map(4)
352     port map(
353       clock  => clock,
354       clear  => clear_reg_s,
355       enable => pronto_med_s,
356       D      => altb_vector,
357       Q      => altb_vector_reg
358     );
359
360   U8_RAM: ram_dist
361     port map(
362       -- Inputs
363       clock => clock,
364       wr    => write_enable,
365       addr  => posicao_s,
366       din   => distancia_hcsr,
367       -- Output
368       dout => distancia_ram
369     );
370
371   -- Conta numero de varreduras: n + 1
372   -- Sendo 2, varre 2 + 1 = 3 vezes
373   U9_CONT: contadorg_m
374     generic map(2)
375     port map(
376       -- Inputs
377       clock   => clock,
378       zera_as => reset,
379       zera_s  => '0',
380       conta   => calibra,
381       -- Outputs
382       Q    => open,
383       fim  => write_stop,
384       meio => open
385     );
386
387   U10_ED: edge_detector
388     port map(
389       -- Inputs
390           clk       => clock,
391           signal_in => last_pos,
392           -- Output
393           output => last_pos_ed
394     );
395
396   U11_RL: registrador_n
397     generic map(1)
398     port map(
399       -- Inputs
400       clock  => clock,
401       clear  => clear_sig,
```

```
402       enable => auth,
403       D      => ligar_vec,
404       -- Output
405       Q => ligar_reg_s
406     );
407
408   U12_RM: registrador_n
409     generic map(1)
410     port map(
411       -- Inputs
412       clock  => clock,
413       clear  => clear_sig,
414       enable => auth,
415       D      => mode_vec,
416       -- Output
417       Q => mode_reg_s
418     );
419
420   U13_RD: registrador_n
421     generic map(12)
422     port map(
423       -- Inputs
424       clock  => clock,
425       clear  => reset,
426       enable => pronto_med_s,
427       D      => distancia_hcsr,
428       -- Output
429       Q => distancia_reg
430     );
431
432   -- Outputs
433   fim_1s             <= fim_1s_s;
434   dado_recebido      <= dado_recebido_s;
435   pronto_med         <= pronto_med_s;
436   posicao_servo      <= posicao_s;
437   distancia          <= distancia_reg;
438   dist_mem           <= distancia_ram;
439   angulo             <= angulo_rom;
440   fim_cal            <= write_stop;
441   alerta_proximidade <= altb_vector_reg(3);
442   dist_mais_sens     <= dist_sens;
443   ligar_reg          <= ligar_reg_s(0);
444   mode_reg           <= mode_reg_s(0);
445
446 end architecture;
```

18.1.4 Password Handler

```
1 library ieee;
2 use ieee.std_logic_1164.all;
3 use ieee.numeric_std.all;
4 use ieee.math_real.all;
5
6 entity pswd_handler is
7     port (
8         -- Input
9         clock    : in std_logic;
```

```
10          reset    : in std_logic;
11          ligar    : in std_logic;
12          mode     : in std_logic;
13          desarmar : in std_logic;
14          pswd_ok  : in std_logic;
15          -- Output
16          wait_pswd : out std_logic;
17          auth      : out std_logic;
18          -- Debug
19          db_event : out std_logic;
20          db_state : out std_logic_vector(3 downto 0)
21      );
22  end entity;
23
24  architecture rtl of pswd_handler is
25      -- Sinais
26      signal reset_uc       : std_logic;
27      signal wait_pswd_s    : std_logic;
28      signal auth_s         : std_logic;
29      signal event_edge     : std_logic;
30      signal timeover_s     : std_logic;
31      signal signal_in_vec  : std_logic_vector(5 downto 0);
32      signal event_edge_vec : std_logic_vector(5 downto 0);
33  begin
34      signal_in_vec(0) <= reset;
35      signal_in_vec(1) <= ligar;
36      signal_in_vec(2) <= not ligar;
37      signal_in_vec(3) <= mode;
38      signal_in_vec(4) <= not mode;
39      signal_in_vec(5) <= desarmar;
40
41      event_edge <= event_edge_vec(5) or event_edge_vec(4) or
        event_edge_vec(3) or event_edge_vec(2) or event_edge_vec(1) or
        event_edge_vec(0);
42
43      GEN_EDGE: for i in 0 to 5 generate
44          UX_ED: entity work.edge_detector (Behavioral)
45              port map(
46                  -- Inputs
47                  clk       => clock,
48                  signal_in => signal_in_vec(i),
49                  -- Output
50                  output => event_edge_vec(i)
51              );
52      end generate;
53
54      U5_CTR: entity work.contadorg_m (comportamental)
55          generic map(500000000) -- 10 s = 500000000
56          port map(
57              -- Inputs
58              clock   => clock,
59              zera_as => auth_s,
60              zera_s  => reset,
61              conta   => wait_pswd_s,
62              -- Outputs
63              Q    => open,
64              fim  => timeover_s,
```

```
65              meio => open
66          );
67
68      U6_UC: entity work.pswd_handler_uc (rtl)
69          port map(
70              -- Input
71              clock    => clock,
72              reset    => reset,
73              event    => event_edge,
74              pswd_ok  => pswd_ok,
75              timeover => timeover_s,
76              -- Output
77              wait_pswd => wait_pswd_s,
78              auth      => auth_s,
79              -- Debug
80              db_state => db_state
81          );
82
83      auth      <= auth_s;
84      wait_pswd <= wait_pswd_s;
85
86      db_event <= event_edge;
87
88  end architecture;
```

18.1.5 Password Handler Control Unit

```
1   library ieee;
2   use ieee.std_logic_1164.all;
3   use ieee.numeric_std.all;
4   use ieee.math_real.all;
5
6   entity pswd_handler_uc is
7       port (
8           -- Input
9           clock    : in std_logic;
10          reset    : in std_logic;
11          event    : in std_logic;
12          pswd_ok  : in std_logic;
13          timeover : in std_logic;
14          -- Output
15          wait_pswd : out std_logic;
16          auth      : out std_logic;
17          -- Debug
18          db_state : out std_logic_vector(3 downto 0)
19      );
20  end entity;
21
22  architecture rtl of pswd_handler_uc is
23      type type_state is
24          (
25              idle, waiting, authorized
26          );
27
28      signal Ecurr : type_state;
29      signal Enext : type_state;
30  begin
```

```
31      -- State memory
32      process (clock)
33      begin
34          if clock'event and clock = '1' then
35              Ecurr <= Enext;
36          end if;
37      end process;
38
39      -- Next state logic
40      process (event, timeover, Ecurr)
41      begin
42          case Ecurr is
43              when idle =>
44                  if event = '1' then
45                      Enext <= waiting;
46                  else
47                      Enext <= idle;
48                  end if;
49
50              when waiting =>
51                  if pswd_ok = '1' then
52                      Enext <= authorized;
53                  elsif timeover = '1' then
54                      Enext <= idle;
55                  else
56                      Enext <= waiting;
57                  end if;
58
59              when authorized => Enext <= idle;
60          end case;
61      end process;
62
63      -- Output logic
64      with Ecurr select
65          wait_pswd <= '1' when waiting, '0' when others;
66
67      with Ecurr select
68          auth <= '1' when authorized, '0' when others;
69
70      -- Debug
71      with Ecurr select
72          db_state <=
73              "0000" when idle,
74              "0001" when waiting,
75              "0010" when authorized;
76  end architecture;
```

18.2 Testbench

18.2.1 Security System Testbench

```
1   library ieee;
2   use ieee.std_logic_1164.all;
3   use ieee.numeric_std.all;
4
5   entity sistema_seguranca_tb is
```

```
6  end entity;
7
8  architecture tb of sistema_seguranca_tb is
9      -- Inputs
10     signal clock_in       : std_logic := '0';
11     signal reset_in       : std_logic := '0';
12     signal ligar_in       : std_logic := '0';
13     signal echo_in        : std_logic := '0';
14     signal dado_serial_in : std_logic := '0';
15     signal mode_in        : std_logic := '0';
16     signal sel_mux_in     : std_logic_vector(1 downto 0) := "00";
17
18     -- Outputs
19     signal trigger_out      : std_logic := '0';
20     signal pwm_out          : std_logic := '0';
21     signal saida_serial_out : std_logic := '0';
22     signal alerta_prox_out  : std_logic := '0';
23     signal calibrando_out   : std_logic := '0';
24     signal alerta_mov_out   : std_logic := '0';
25
26     -- Controle do clock
27     signal keep_simulating : std_logic := '0';
28     constant clockPeriod   : time      := 20 ns;
29
30     -- Array de casos de teste
31   type caso_teste_type is record
32     id    : natural;
33     tempo : integer;
34   end record;
35
36   type casos_teste_array is array (natural range <>) of
       caso_teste_type;
37   constant casos_teste : casos_teste_array :=
38     (
39       (1, 100),
40       (2, 231),
41             (3, 300),
42             (4, 392),
43             (5, 492),
44             (6, 392),
45             (7, 300)
46             -- (8, 231),
47             -- (9, 100),
48             -- (10, 231),
49             -- (11, 300),
50             -- (12, 392),
51             -- (13, 492),
52             -- (14, 392),
53             -- (15, 300),
54             -- (16, 231),
55             -- (17, 100),
56             -- (18, 231),
57             -- (19, 300),
58             -- (20, 330)
59     );
60
61   signal larguraPulso: time := 1 ns;
```

```
62
63 begin
64
65     -- Gerador de clock
66     clock_in <= (not clock_in) and keep_simulating after
       clockPeriod/2;
67
68     -- Instancia
69     DUT: entity work.sistema_seguranca (sistema_seguranca_arch)
70         port map(
71             -- Inputs
72             clock       => clock_in,
73             reset       => reset_in,
74             ligar       => ligar_in,
75             echo        => echo_in,
76             dado_serial => dado_serial_in,
77             mode        => mode_in,
78             sel_mux     => sel_mux_in,
79             -- Outputs
80             trigger            => trigger_out,
81             db_trigger         => open,
82             db_echo            => open,
83             pwm                => pwm_out,
84             saida_serial       => saida_serial_out,
85             saida_serial_ch    => open,
86             saida_serial_mqtt  => open,
87             pwm_ch             => open,
88             alerta_proximidade => alerta_prox_out,
89             alerta_prox_mqtt   => open,
90             db_transmitir      => open,
91             db_medir           => open,
92             calibrando         => calibrando_out,
93             alerta_mov         => alerta_mov_out,
94             db_mode            => open,
95             db_fim_2s          => open,
96             display0           => open,
97             display1           => open,
98             display2           => open,
99             display3           => open,
100            display4           => open,
101            display5           => open
102        );
103
104    -- Estimulo
105    stim: process is
106    begin
107        assert false report "Inicio das simulacoes" severity note;
108    keep_simulating <= '1';
109
110    ---- valores iniciais ----------------
111    echo_in <= '0';
112        mode_in <= '0';
113
114    ---- inicio: reset ----------------
115    wait for 2*clockPeriod;
116    reset_in <= '1';
117    wait for 2 us;
```

```
118     reset_in <= '0';
119     wait until falling_edge(clock_in);
120 
121     ---- espera de 100us
122     wait for 100 us;
123 
124         wait until falling_edge(clock_in);
125         ligar_in <= '1';
126 
127     ---- loop pelos casos de teste
128     for i in casos_teste'range loop
129       -- 1) determina largura do pulso echo
130       assert false report "Caso de teste " & integer'image(
    casos_teste(i).id) & ": " &
131         integer'image(casos_teste(i).tempo) & "us" severity note;
132       larguraPulso <= casos_teste(i).tempo * 1 us; -- caso de teste
     "i"
133 
134       -- 2) espera por 400us (tempo entre trigger e echo)
135       wait for 400 us;
136 
137       -- 3) gera pulso de echo (largura = larguraPulso)
138       echo_in <= '1';
139       wait for larguraPulso;
140       echo_in <= '0';
141 
142       -- 4) espera entre casos de tese
143       wait for 20 ms;
144 
145     end loop;
146 
147         wait for 500 ms;
148 
149     ---- final dos casos de teste da simulacao
150     assert false report "Fim das simulacoes" severity note;
151     keep_simulating <= '0';
152 
153     wait; -- fim da simula  o: aguarda indefinidamente (n  o
    retirar esta linha)
154 
155     end process;
156 
157 end architecture;
```

18.3 Dashboard Processing

```
1  //-------------------------Laboratorio Digital II
       ----------------------------
2  // Projeto: QuietWatch (T3BB5)
3  // Gabriel Pereira de Carvalho
4  // Ot vio Felipe de Freitas
5  // Willian Abe Fukushima
6  //
       ------------------------------------------------------------------------------

7  // Para interagir com interface, s o utilizados comandos de
       teclado (PUBLISH)
```

```
8  // l = entrada ligar
9  // r = entrada reset
10 // m = entrada mode
11 // d = entrada desarmar
12 //
       ------------------------------------------------------------------------------


13
14 //Bibliotecas Java
15 import java.io.IOException;
16 import java.time.format.DateTimeFormatter;
17 import java.time.LocalDateTime;
18 //Bibliotecas Processing
19 import mqtt.*;
20 import ddf.minim.*;
21
22 //------------------------------ Conex o MQTT
       ----------------------------------
23 String user   = "grupo1-bancadaB5";
24 String passwd = "L%40Bdygy1B5";      // manter %40
25 String broker = "3.141.193.238";
26 String port   = "80";
27 MQTTClient client;
28 String clientID;
29 //
       ------------------------------------------------------------------------------


30
31 //-------------------------------- Variaveis
       ---------------------------------
32 int whichKey = -1;  // variavel mantem tecla acionada
33 Boolean estado_ligar = false;
34 Boolean estado_mode = false;
35 Boolean estado_reset = false;
36 Boolean estado_selmux0 = false;
37 Boolean estado_selmux1 = false;
38 Boolean estado_alerta = false;
39 Boolean estado_calibrando = false;
40 Boolean estado_desarmar = false;
41 Boolean estado_senha = false;
42 String senha = "";
43 PFont myFont;
44 int largura = 960;
45 int altura = 600;
46 PrintWriter QuietWatch_logger;
47 Minim minim;
48 AudioPlayer som_calibrando, som_alarme;
49 //
       ------------------------------------------------------------------------------


50
51 //------------------------- Setup: executado uma vez
       --------------------------
52 void setup() {
53   // Tamanho do quadro
54   size(960, 600);
55   myFont = loadFont("myfont.vlw");
```

```
56    smooth();
57    //setup PrintWriter
58    QuietWatch_logger = createWriter("QuietWatch_logs_" + day() + "_"
         + month() + "_" + year() + "_horario_" + hour() + "_" + minute
        () + "_" + second() + ".txt");
59    QuietWatch_logger.println("Setup Processing " + day() + "/" +
        month() + "/" + year() + " horario " + hour() + ":" + minute()
        + ":" + second());
60    QuietWatch_logger.flush();
61    // setup Minim
62    minim = new Minim(this);
63    som_calibrando = minim.loadFile("som_calibrando.mp3");
64    som_alarme = minim.loadFile("som_alarme.mp3");
65    // Conectar com MQTT
66    client = new MQTTClient(this);
67    clientID = new String("labead-mqtt-processing-" + random(0,100));
68    println("clientID=" + clientID);
69    client.connect("mqtt://" + user + ":" + passwd + "@" + broker + "
        :" + port, clientID, false);
70    // Garantir sinais zerados
71    client.publish(user + "/E0home", "0");//reset
72    client.publish(user + "/E1home", "0");//ligar
73    client.publish(user + "/E2home", "0");//mode
74    client.publish(user + "/E3home", "0");//desarmar
75  }
76  //
        ------------------------------------------------------------------------


77
78  //----------------------- Draw: executado continuamente
        -----------------------
79  void draw() {
80      cursor(HAND);
81      textFont(myFont);
82      background(#B6BCB3);
83      // chama funcoes para desenhar painel do sistema de seguran a
84      drawCabecalho();
85      drawQuadro();
86  }
87  //
        ------------------------------------------------------------------------


88
89
90  void drawCabecalho() {
91      textAlign(CENTER);
92      textSize(30);
93      fill(0);
94      text("Controle os Widgets com o teclado", 10, 10, largura,
        altura);
95      textAlign(CENTER);
96      textSize(15);
97      fill(0);
98      text("r : E0 (reset)", 10, 40, largura, altura);
99      textAlign(CENTER);
100     textSize(15);
101     fill(0);
```

```
102     text("l: E1 (ligar)", 10, 55, largura, altura);
103     textAlign(CENTER);
104     textSize(15);
105     fill(0);
106     text("m: E2 (modo 0 'em casa' ou 1 'fora de casa')", 10, 70,
        largura, altura);
107     textAlign(CENTER);
108     textSize(15);
109     fill(0);
110     text("d: E3 (desarmar)", 10, 85, largura, altura);
111
112 }
113 void drawQuadro(){
114   stroke(#300A8B);
115   fill(#868489);
116   rect(250, 100, 500, 410);
117 //--------------------------- Widgets para Inputs
        ----------------------------
118   if(estado_ligar){//l de ligar
119     fill(#FF0303);
120     ellipse(350, 150, 70, 70);
121     textSize(12);
122     fill(0);
123     text("on", 350, 155);
124   }else{
125     fill(#B6BCB3);
126     ellipse(350, 150, 70, 70);
127     textSize(12);
128     fill(0);
129     text("off", 350, 155);
130   }
131   if(estado_mode){// m de mode
132     fill(#FF0303);
133     ellipse(350, 250, 70, 70);
134     textSize(11);
135     fill(0);
136     text("Fora de Casa", 350, 255);
137   }else{
138     fill(#B6BCB3);
139     ellipse(350, 250, 70, 70);
140     textSize(12);
141     fill(0);
142     text("Em Casa", 350, 255);
143   }
144   if(estado_reset){// r de reset
145     fill(#FF0303);
146     ellipse(350, 350, 70, 70);
147     textSize(12);
148     fill(0);
149     text("reset", 350, 355);
150   }else{
151     fill(#B6BCB3);
152     ellipse(350, 350, 70, 70);
153     textSize(12);
154     fill(0);
155     text("reset", 350, 355);
156   }
```

```
157   if(estado_desarmar){// d de desarmar(E3)
158     fill(#FF0303);
159     ellipse(350, 450, 70, 70);
160     textSize(10);
161     fill(0);
162     text("Aguardando", 350, 455);//E3 = 1
163   }else{
164     fill(#B6BCB3);
165     ellipse(350, 450, 70, 70);
166     textSize(12);
167     fill(0);
168     text("Desarmar", 350, 455);//E3 = 0
169   }
170
171   //
        ------------------------------------------------------------------------------

172   stroke(255,0,0);
173   strokeWeight(5);
174   line(500,100,500,510);
175   //-------------------------- Widgets para Outputs
        ----------------------------
176   if(estado_alerta){
177     stroke(#300A8B);
178     fill(#FF0303);
179     ellipse(600, 150, 70, 70);
180     textSize(12);
181     fill(0);
182     text("alerta on", 600, 155);
183   }else{
184     stroke(#300A8B);
185     fill(#00FF12);
186     ellipse(600, 150, 70, 70);
187     textSize(12);
188     fill(0);
189     text("alerta off", 600, 155);
190   }
191   if(estado_calibrando){
192     stroke(#300A8B);
193     fill(#FF0303);
194     ellipse(600, 250, 70, 70);
195     textSize(12);
196     fill(0);
197     text("calibrando...", 600, 255);
198   }else{
199     //stroke(#300A8B);
200     //if(estado_ligar){
201     //  fill(#00FF12);
202     //  ellipse(600, 250, 70, 70);
203     //  textSize(12);
204     //  fill(0);
205     //  text("calibrado", 600, 255);
206     //}else{
207     //  fill(#B6BCB3);
208     //  ellipse(600, 250, 70, 70);
209     //}
210   }
```

```
211   if(estado_senha){
212     stroke(#300A8B);
213     fill(#FF0303);
214     ellipse(600, 370, 100, 100);
215     textSize(12);
216     fill(0);
217     text("INSERIR SENHA", 600, 375);
218   }else{
219     //stroke(#300A8B);
220     //fill(#B6BCB3);
221     //ellipse(600, 370, 100, 100);
222     //textSize(12);
223     //fill(0);
224     //text("", 600, 155);
225   }
226   //
        ------------------------------------------------------------------------------

227   String dispTime = "Hora: " + hour() + ":" + minute() + ":" +
        second();
228   fill(0);
229   rect(605, 485, 85, 20);
230   fill(#B6BCB3);
231   text(dispTime, 650, 500);
232 }
233
234 //------------------- keyPressed: processa entrada por teclado
        ------------------
235 // funcao keyPressed - processa tecla acionada
236 void keyPressed() {
237   whichKey = key;
238   if(estado_senha){
239     String key_char = str((char) whichKey);
240     senha += key_char;
241     if(senha.length() == 4){
242       println("SENHA ENVIADA " + senha);
243       QuietWatch_logger.println("SENHA ENVIADA  " + senha);
244       QuietWatch_logger.flush();
245       client.publish(user + "/RXhome", senha);
246       senha = "";
247     }
248   }else{
249     if(whichKey == 114){
250       estado_reset = !estado_reset;
251       if(estado_reset){
252         client.publish(user + "/E0home", "1");
253         QuietWatch_logger.println("Reset  " + hour() + ":" + minute
        () + ":" + second());
254         QuietWatch_logger.flush();
255       }else{
256         client.publish(user + "/E0home", "0");
257       }
258     }
259     if(whichKey == 108){
260       estado_ligar = !estado_ligar;
261       if(estado_ligar){
262         client.publish(user + "/E1home", "1");
```

```
263         QuietWatch_logger.println("Ligar  " + hour() + ":" + minute
    () + ":" + second());
264         QuietWatch_logger.flush();
265       }else{
266         client.publish(user + "/E1home", "0");
267       }
268     }
269     if(whichKey == 109){
270       estado_mode = !estado_mode;
271       if(estado_mode){
272         client.publish(user + "/E2home", "1");
273         QuietWatch_logger.println("Modo Fora de Casa  " + hour() +
    ":" + minute() + ":" + second());
274         QuietWatch_logger.flush();
275       }else{
276         client.publish(user + "/E2home", "0");
277         QuietWatch_logger.println("Modo Em Casa  " + hour() + ":" +
     minute() + ":" + second());
278         QuietWatch_logger.flush();
279       }
280     }
281     if(whichKey == 100){
282       estado_desarmar = !estado_desarmar;
283       if(estado_mode){
284         client.publish(user + "/E3home", "1");
285         QuietWatch_logger.println("Desarmado  " + hour() + ":" +
    minute() + ":" + second());
286         QuietWatch_logger.flush();
287       }else{
288         client.publish(user + "/E3home", "0");
289         QuietWatch_logger.println("Armado  " + hour() + ":" +
    minute() + ":" + second());
290         QuietWatch_logger.flush();
291       }
292     }
293   }
294 }
295 //
    ------------------------------------------------------------------------


296
297 //---------------------------- Fun   es para MQTT
    ----------------------------
298 void clientConnected() {
299   println("cliente conectado");
300   client.subscribe(user + "/E0home");
301   client.subscribe(user + "/E1home");
302   client.subscribe(user + "/E2home");
303   client.subscribe(user + "/E3home");
304   client.subscribe(user + "/S0home");//alerta movimento
305   client.subscribe(user + "/S1home");//calibrando
306   client.subscribe(user + "/S2home");//senha
307 }
308
309 void messageReceived(String topic, byte[] payload) {
310   String dados = new String(payload);
311
```

```
312   if(topic.endsWith("E0home")){
313     if(Integer.parseInt(dados) == 1){
314       QuietWatch_logger.println("Recebido Reset  " + hour() + ":" +
       minute() + ":" + second());
315       QuietWatch_logger.flush();
316       estado_reset = true;
317     }else{
318       estado_reset = false;
319     }
320   }
321
322   if(topic.endsWith("E1home")){
323     if(Integer.parseInt(dados) == 1){
324       QuietWatch_logger.println("Recebido Ligar  " + hour() + ":" +
       minute() + ":" + second());
325       QuietWatch_logger.flush();
326       estado_ligar = true;
327     }else{
328       estado_ligar = false;
329     }
330   }
331
332   if(topic.endsWith("E2home")){
333     if(Integer.parseInt(dados) == 1){
334       QuietWatch_logger.println("Recebido Modo Fora de Casa  " +
     hour() + ":" + minute() + ":" + second());
335       QuietWatch_logger.flush();
336       estado_mode = true;
337     }else{
338       QuietWatch_logger.println("Recebido Modo Em Casa  " + hour()
     + ":" + minute() + ":" + second());
339       QuietWatch_logger.flush();
340       estado_mode = false;
341     }
342   }
343   if(topic.endsWith("E3home")){
344     if(Integer.parseInt(dados) == 1){
345       QuietWatch_logger.println("Recebido Desarmado  " + hour() + "
     :" + minute() + ":" + second());
346       QuietWatch_logger.flush();
347       estado_desarmar = true;
348     }else{
349       QuietWatch_logger.println("Recebido Armado  " + hour() + ":"
     + minute() + ":" + second());
350       QuietWatch_logger.flush();
351       estado_desarmar = false;
352     }
353   }
354
355   if(topic.endsWith("S0home")){//mensagem veio de S0 (alerta_mov)
356     if(Integer.parseInt(dados) == 1){//alerta esta ativado
357       DateTimeFormatter dtf = DateTimeFormatter.ofPattern("yyyy/MM/
     dd HH:mm:ss");
358       LocalDateTime now = LocalDateTime.now();
359       if(estado_alerta == false){
360         println("ALERTA DE MOVIMENTO : " + dtf.format(now));
```

```
361         QuietWatch_logger.println("ALERTA DE MOVIMENTO  " + hour()
    + ":" + minute() + ":" + second());
362         QuietWatch_logger.flush();
363         if(estado_mode){
364           som_alarme.play();
365         }
366         estado_alerta = true;
367       }
368     }else{
369       estado_alerta = false;
370       som_alarme.pause();
371     }
372   }
373
374   if(topic.endsWith("S1home")){//mensagem veio de S1 (calibrando)
375     if(Integer.parseInt(dados) == 1){//alerta esta ativado
376       DateTimeFormatter dtf = DateTimeFormatter.ofPattern("yyyy/MM/
    dd HH:mm:ss");
377       LocalDateTime now = LocalDateTime.now();
378       if(estado_calibrando == false){
379         println("Calibrando : " + dtf.format(now));
380         QuietWatch_logger.println("Calibrando  " + hour() + ":" +
    minute() + ":" + second());
381         QuietWatch_logger.flush();
382         estado_calibrando = true;
383       }
384     }else{
385       estado_calibrando = false;
386     }
387   }
388
389   if(topic.endsWith("S2home")){//mensagem veio de S2
390     if(Integer.parseInt(dados) == 1){
391       DateTimeFormatter dtf = DateTimeFormatter.ofPattern("yyyy/MM/
    dd HH:mm:ss");
392       LocalDateTime now = LocalDateTime.now();
393       if(estado_senha == false){
394         println("Senha : " + dtf.format(now));
395         QuietWatch_logger.println("Inserir Senha  " + hour() + ":"
    + minute() + ":" + second());
396         QuietWatch_logger.flush();
397         estado_senha = true;
398       }
399     }else{
400       estado_senha = false;
401     }
402   }
403 }
404 void connectionLost() {
405   println("conexao perdida");
406 }
407 //
    ------------------------------------------------------------------------
```