IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

Impact Factor 8.414 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 6, June 2025

DOI: 10.17148/IJIREEICE.2025.13603

A novel, heart diagnosis tool, based on Electro-cardiogram (ECG), using VHDL and FPGAs

Dr Evangelos I. Dimitriadis[1], Leonidas Dimitriadis[2]

Department of Computer, Informatics and Telecommunications Engineering, International Hellenic University, End of Magnisias Str, 62124 Serres Greece.[1]

Undergraduate student, Department of Information and Electronic Engineering, International Hellenic University, 57400, Sindos Thessaloniki, Greece. [2]

Abstract: An FPGA-based system is presented here, capable of simultaneous measuring electrocardiogram (ECG) input values and display them in seven-segment displays, while it also derives a heart diagnosis, based on the above ECG. Heart diagnosis activates corresponding LED alarm system, providing information about several heart problems such as hypercalcemia, hypocalcemia, hyperkalemia, coronary ischemia, bundle-branch block and AV node block, all of them derived from critical time values of the ECG. Our system has also the ability of calculating and presenting in seven-segment displays, heart beat rate in pulses/min, with simultaneous activation of corresponding alarm system, which is responsible of lighting three respective heart beat rate LEDs, for vradycardia, normal heart beat rate and tachycardia. Additionally, buzzer alarm and all FPGA board LEDs are also simultaneously activated, if pulses/min values are less than 60 (vradycardia) or higher than 100 (tachycardia). The system uses DE10-Lite FPGA board with an HW 827 sensor connected to it. The above sensor output is also connected to a blinking LED system, in order to have visual information of heart beat rate. Our system can work with a variety of heart pulse sensors and it is clear that deriving a successful heart diagnosis requires an electrocardiogram recording that is as accurate as possible. The system can also be combined with IoT technology, offering doctors the ability to remotely monitor patients online.

Keywords: Heart beat rate, Sensors, Heart diagnosis, VHDL, Buzzer, LEDs

I. INTRODUCTION

FPGAs have the main advantage of combining software and hardware, thus enabling hardware programming for a series of applications. The most used languages for FPGAs' programing are VHDL and Verilog and VHDL is the one used in our work.

An interesting application field of FPGAs is heart beat sensoring and monitoring. We found out that in spite of all work done concerning various FPGAs applications, there are a few projects [1-7] involved with heart systems. Presented works use expensive and complicated systems and the main problem they try to solve is recording, filtering and presenting a sufficient electrocardiogram (ECG). Our system uses electrocardiogram input values, in order to derive a heart diagnosis, valuable for patients monitoring. It uses several alarm systems for heart diagnosis, heart beat rate characterization and simultaneous buzzer activation, as well as heart beat visualization via a blinking LED system. Another benefit of our system is that it can work with a variety of heart sensors and also its cost is remarkably low.

II. DESIGN OVERVIEW AND OPERATION OF THE SYSTEM

Figure1 presents device overview and operational units of our system, using FPGA DE10-Lite board, while Figure 2 presents circuit diagram of the system.

It is obvious from both of the above figures that our system, except from DE10-Lite FPGA board, contains also some basic circuit parts. The first one is heart beat LED blinking unit used for providing a visualized view of heart beat input. Next to it there is heart beat rate LEDs unit used for lighting respective LEDs, depending on pulses/min values (vradycardia, normal heart beat rate and tachycardia). This unit has also a buzzer circuit containing a transistor and diode and it is connected in I/O pins of the FPGA

IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

Impact Factor 8.414 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 6, June 2025

DOI: 10.17148/IJIREEICE.2025.13603

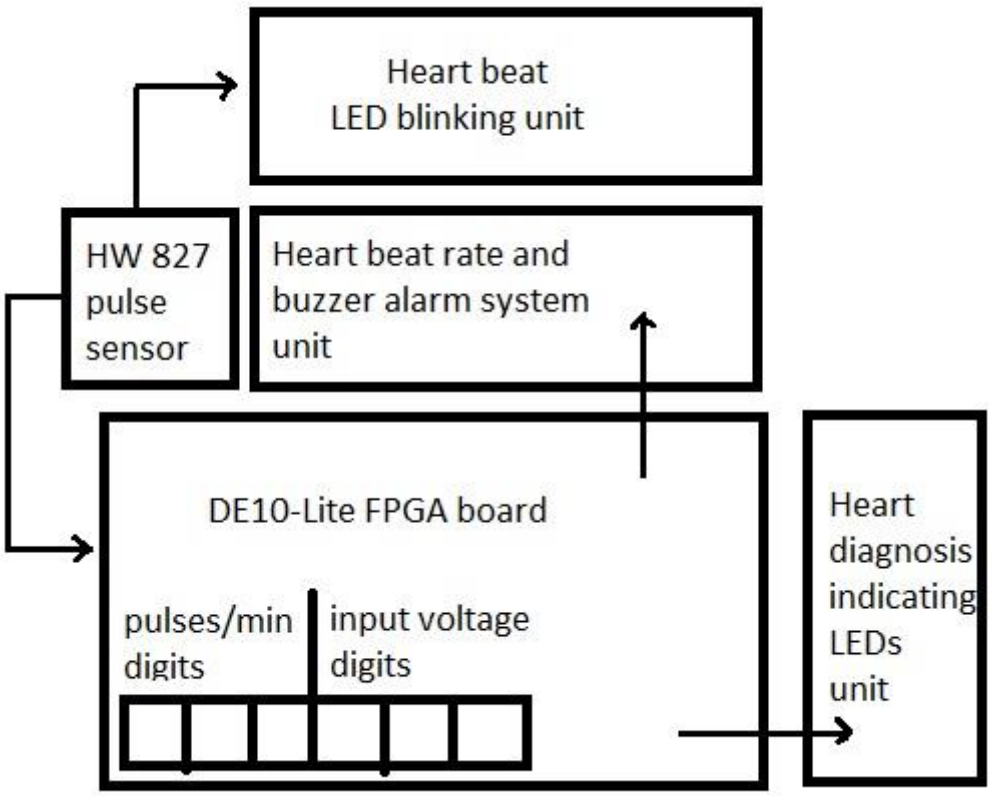

Figure 1: Device overview and operational units of our system.

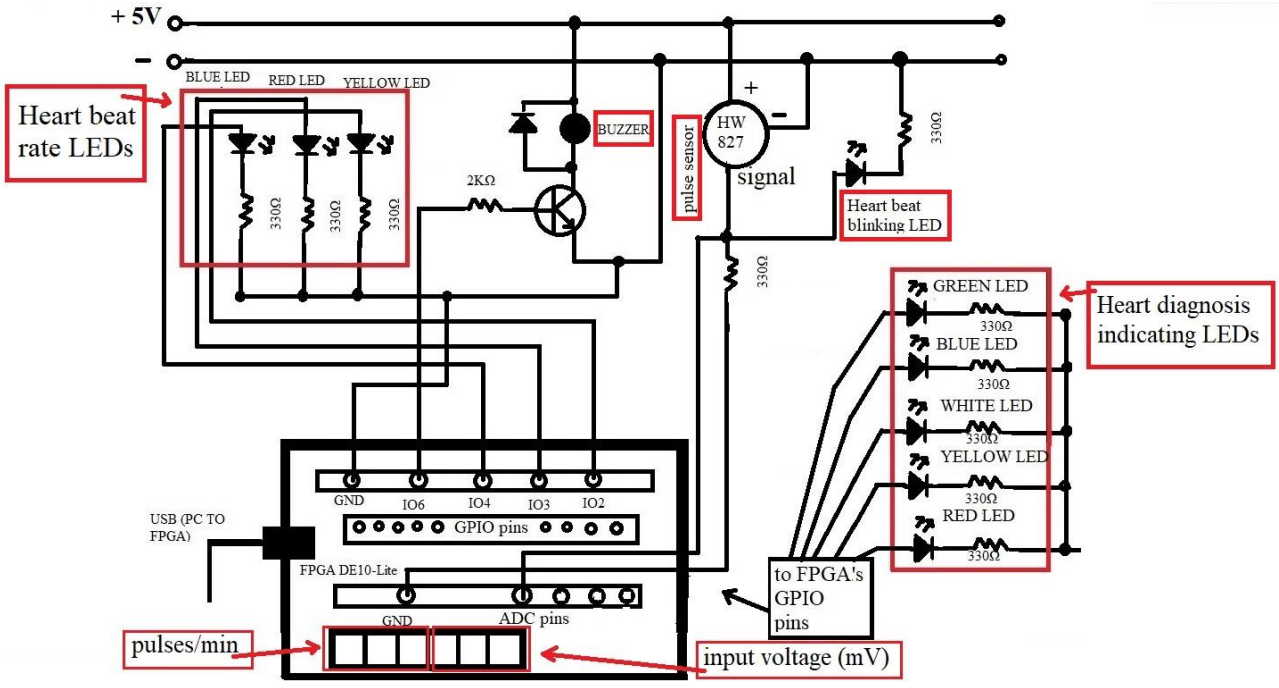

Figure 2: Circuit diagram of our system

IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

Impact Factor 8.414 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 6, June 2025

DOI: 10.17148/IJIREEICE.2025.13603

board. It is the circuit that controls buzzer's operation. The transistor is used for amplifying signal bit 1 sent by the I/O pins of FPGA board, in order to provide sufficient voltage supply for buzzer operation. Buzzer is activated if heart beat rate is out of normal values of 60-100 pulses/min.

Heart diagnosis indicating LEDs unit is another very important part of our system. Depending on ECG input values the above unit lights up respective LEDs, thus giving us information about the existence of several heart problems.

DE10-Lite FPGA board used here offers its seven-segment displays for presenting pulses/min and input ECG voltage values.

HW 827 heart beat pulse sensor, acts as main input unit in conjunction with FPGA's clock which provides us with the second very important parameter of an ECG, which is time. Figures 3 and 4 present our system and its units, in function condition.

Our system starts operating as soon as power supply +5V is applied to all circuits and the VHDL program is sent via USB Blaster interface, to FPGA chip, reading at first input voltages from HW 827 pulse sensor, with simultaneous start of time measurement. The analogue input voltages are converted to digital and presented in seven-segment displays. The system receives input voltage values periodically, ensuring continuous voltage change monitoring.

Consequently, units of heart beat blinking LED, heart beat rate LEDs and heart diagnosis indicating LEDs, are put in use. The first unit is not receiving an output bit from FPGA I/O pins and no VHDL program is necessary for it to function. The other two units receive bit 1 for their different LEDs in order to light them ON. In heart beat rate unit we use three LEDs, blue, red and yellow and they transit to ON condition if calculated pulses/min<60, pulses/min>100 and 60<=pulses/min<=100, respectively. As we mentioned above, if pulses/min value is not normal, buzzer alarm and all FPGA board LEDs, also transit to ON state.

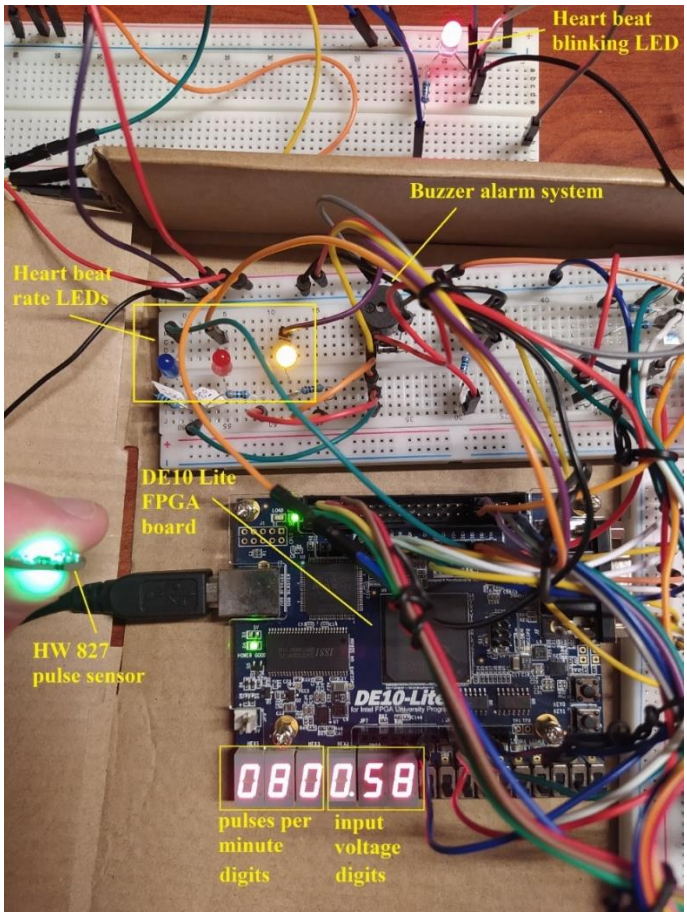

Figure 3: Normal function condition of our system, presenting pulses /min and input voltage values with simultaneous heart beat blinking LED working, according to HW 827 pulse sensor input.

IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

Impact Factor 8.414 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 6, June 2025

DOI: 10.17148/IJIREEICE.2025.13603

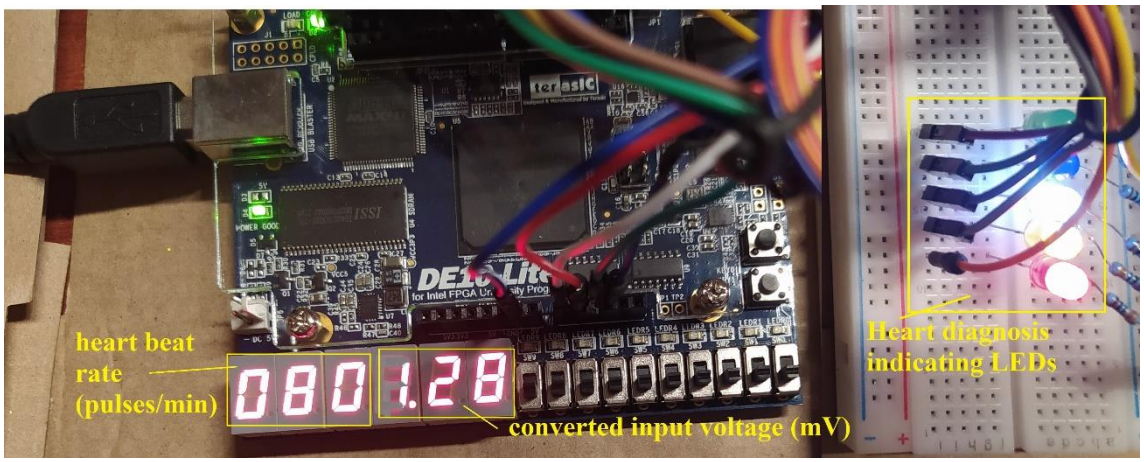

Figure 4: Heart diagnosis part of our system, presenting existing heart problems, based on ECG input.

Heart diagnosis indicating LEDs unit, is the third of the above three units which starts operating. It uses five LEDs, green, blue, white, yellow and red. Each one of them must receive bit 1 from FPGA I/O pins, in order to light ON. If green LED is ON it means that we have a normal electrocardiogram (ECG) input. Each one of the other four LEDs, corresponds to one of the heart problems that will be presented below.

All the above controls are periodically operated as long as the system is at the ON state. The system goes to OFF state if external circuit voltage supply is OFF or if FPGA board is unplugged from USB Blaster, or both of them.

III. PROGRAMING THE SYSTEM

We used Quartus Prime Lite Edition 21.1.1 to create the VHDL programs of our system. It must be mentioned here that before proceeding with the VHDL programming of our system, we had to set a series of parameters controlling the operation of DE10-Lite FPGA's Analog to Digital Converter (ADC). This converter plays a very important role in the whole system operation, since it converts the analogue input voltages from HW 827 sensor connected to FPGA board to digital values, acting as main input of the system. The files created by the above ADC parameters setting are imported into the final project of our system.

We present below in Figure 5 typical sinus rhythm electrocardiogram (ECG), with critical time values used in our study, shown in red.

An ECG includes a number of very important parts, such as P wave, QRS complex, T wave and also critical time intervals, such as PR interval, PR segment, ST segment and QT interval, all of them providing precious information about heart function.

It is obvious that an ECG diagram presents voltage values (y-axis) versus time values (x-axis).

IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

Impact Factor 8.414 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 6, June 2025

DOI: 10.17148/IJIREEICE.2025.13603

Table 1 shows basic ECG parameter values [2,6,7] appeared in bibliography.

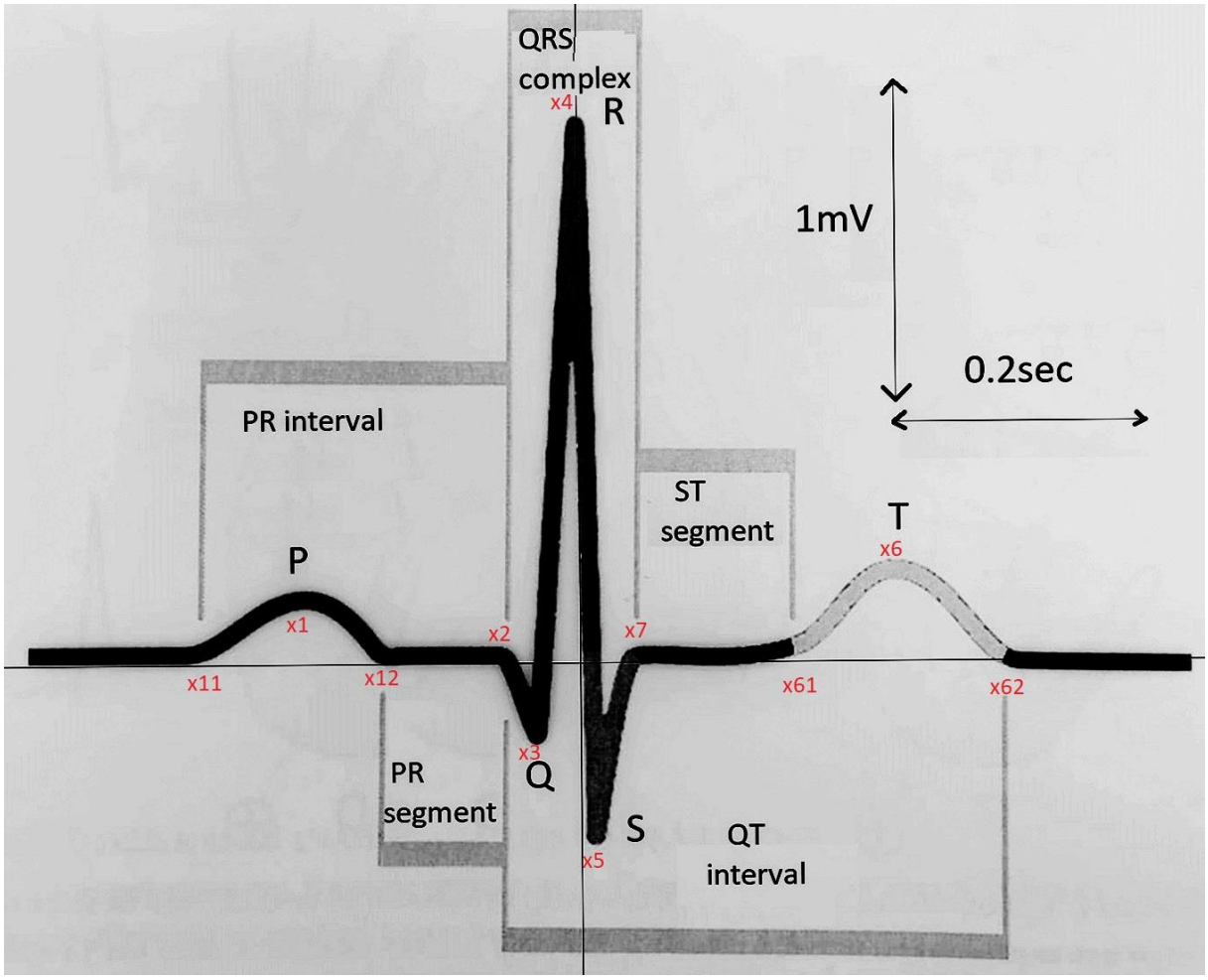

Figure 5: Typical sinus rhythm electrocardiogram (ECG), with critical time values used in our study, shown in red.

Table 1: ECG parameter values

P wave	PR interval	PR segment	QRS complex	ST segment	QT interval	T wave
0.11sec, 0.1-0.25 mV	0.12-0.2 sec	0.06-0.1 sec	0.09-0.1 sec, 1-1.6 mV	0.10-0.15 sec	0.35-0.44 sec	0.1-0.15 sec, 0.5mV

All the above parameters play critical role in heart function and in conjunction with data presented in Table 2, are the basis of our VHDL program controlling the system presented in this work.

Table 2 shows several heart problems that arise from specific ECG parameter values. It also presents corresponding LEDs that light ON, in order to inform us about respective heart problems.

IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

Impact Factor 8.414 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 6, June 2025

DOI: 10.17148/IJIREEICE.2025.13603

Table 2: Heart problems related to ECG parameter values and corresponding system LEDs ON

Short QT interval-Long QT interval	Flat or inverted T waves	Increased QRS	Increased PR
Hypercalcemia, hyper-kalemia-hypocalcemia	Coronary ischemia, hypokalemia, left ven-tricular hypertrophy	Bundle-branch block. Depolarization of ven-tricles and triggering main pumping contrac-tions, delays	AV node block, which results in long delay of AV node to allow fill-ing of ventricles
Red led alarm	Yellow led alarm	White led alarm	Blue led alarm

The main idea of our program is to distinguish specific ECG input voltage values and their corresponding time values, in order to calculate critical time intervals shown in Table 2, thus resulting in a heart diagnosis. Consequently corresponding LEDs light ON. If we distinguish, for example P wave input voltage value, we can create in a VHDL process a respective time variable x1 as shown in Figure 5. Taking x1 as data and combining it with Table 1 in which P wave has a time duration of 0.11 sec, with x1 being in the middle of this time period, we can calculate time values of x11 and x12, also shown in Figure 5.

A similar method is used in order to distinguish T wave and its x6, x61 and x62 time values. The R wave of QRS complex is easy to be distinguished because it corresponds to the higher input voltage value, as presented in Table 1. Since R wave is almost in the middle of a normal sinus heart pulse, we can double its corresponding time value x4 and calculate the period of one heart pulse in seconds, thus making it easy to produce the number of pulses/min.

A correction factor may be needed for the final value of pulses/min, depending on the heart pulse sensor that we use as main ECG values input.

IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

Impact Factor 8.414 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 6, June 2025

DOI: 10.17148/IJIREEICE.2025.13603

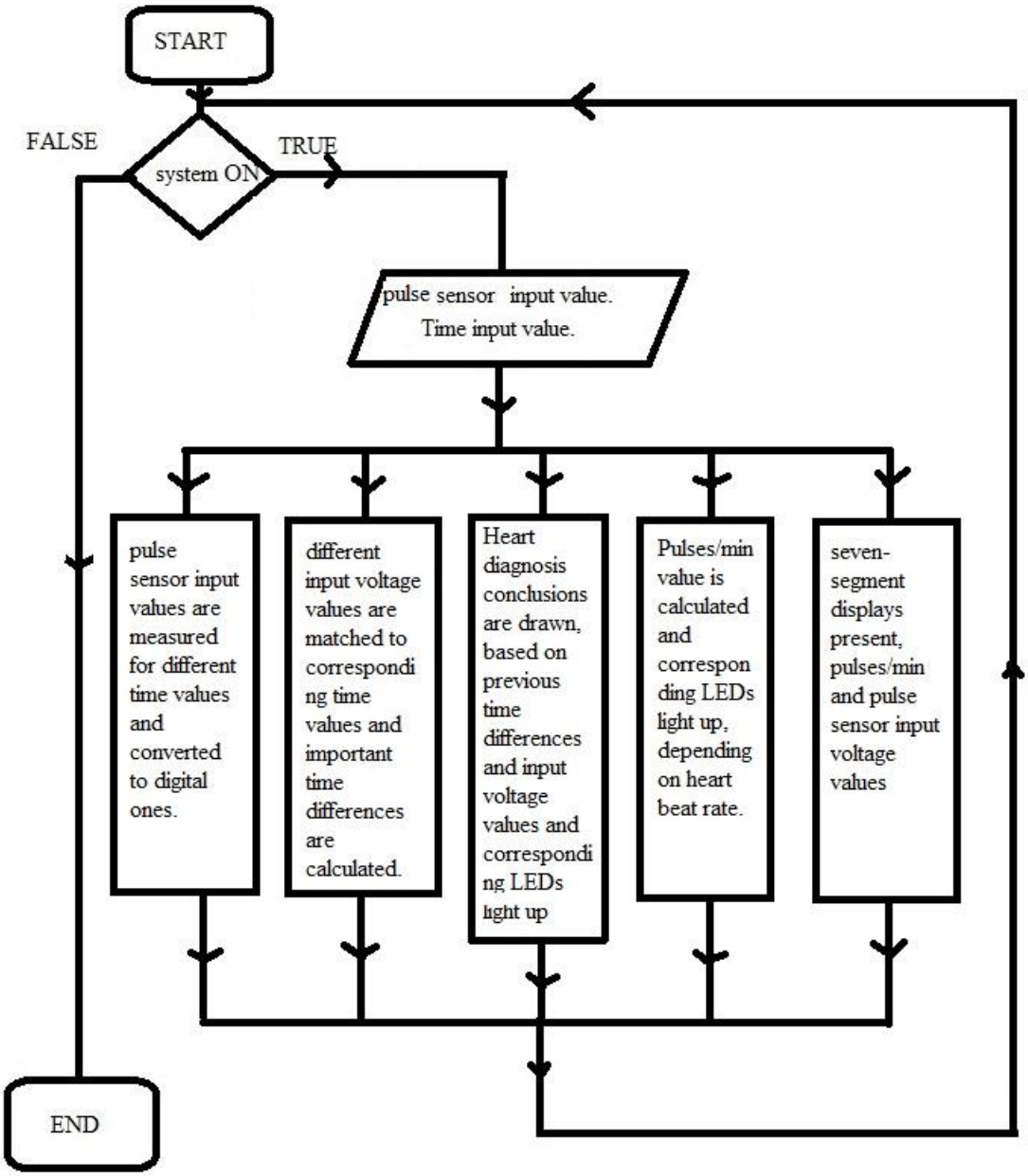

Figure 6: Flowchart diagram, presenting main functions-processes of our system.

The overall program is shown in the appendix of this paper and Figure 6 shows a flowchart diagram, presenting main functions of our system and contains the main algorithmic procedures used here.

Since heart beat rate and the time intervals shown in Table 2 are calculated, we use a number of VHDL processes in order to display pulses/min value and light ON corresponding LEDs for vradycardia, normal heart beat rate and tachy-cardia, as mentioned above.

LEDs that present heart problems shown in Table 2, also become active, depending on respective VHDL process used with ECG input values.

IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

Impact Factor 8.414 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 6, June 2025

DOI: 10.17148/IJIREEICE.2025.13603

Another VHDL process in our program, controls the activation of all FPGA board LEDs and buzzer sounding, if pulses/min values are less than 60 or higher than 100, as shown in the above Figure 6.

Figure 7 presents an ECG-like diagram produced by HW 827 pulse sensor, used in this work with input voltage values converted to mV, in order to fit a classic ECG diagram. The above values were verified by a digital voltmeter.

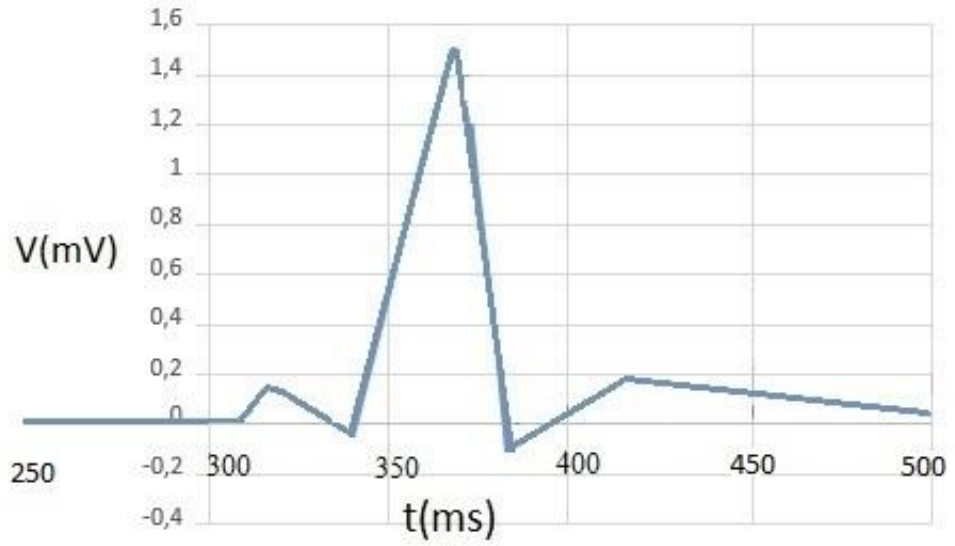

Figure 7: An ECG-like diagram produced by HW 827 pulse sensor, used in this work.

The above ECG diagram was used as an interesting test for our system's reliability and especially for the VHDL algorithms used here. We observe very good agreement and matching between VHDL algorithms and the results which came out of our system's different control units. The R wave corresponds to an input voltage value of 1.5 mV and time value is near 370 ms, which is the value of x4 time variable mentioned above and presented in Figure 5. It results in a pulses/min value of about 80, which is the value appearing in seven-segment displays of the FPGA board shown in Figures 3 and 4.

The heart diagnosis indicating LEDs unit presented in Figure 4, has three LEDs at the ON state. White, yellow and red LEDs are ON indicating corresponding heart problems presented in Table 2. White LED ON indicates increased QRS complex and we see that Figure 7 shows a QRS time interval value from 310 ms to 420 ms, thus overcoming with marginal exceedance the value of 100 ms= 0.1 s presented in Table 1.

Yellow LED ON indicates flat or inverted T waves and it is obvious from Figure 7 that we obtain nearly flat T waves. Finally red LED ON indicates in our case, long QT interval which holds true, since we have no T wave in Figure 7, for limiting QT interval.

As we mentioned earlier, a successful ECG recording could lead to more accurate heart diagnosis and heartbeat rate calculation, thus sensor system used for ECG monitoring plays decisive role for obtaining valid results.

IV. CONCLUSIONS

A novel FPGA-based system is presented here, which manages to monitor an electrocardiogram (ECG) and activate several units, with their behavior based on the ECG. The system uses VHDL language and a number of processes which give the abilities of displaying input voltage values, calculating and displaying heart beat rate and its characterization by activating specific LEDs and also deriving a heart diagnosis using for output five LEDs, presenting corresponding heart problems. Additionally our system provides a visualized view of heart beat rate by using another LED connected to HW 827 pulse sensor. The above sensor acts as the main input for our system. In case that heart beat rate is out of normal values (60-100 pulses/min), buzzer alarm and all FPGA board LEDs are activated. It is obvious that we must have accurate and clear ECG data, in order to obtain valid results from our system. The system can work with a variety of pulse sensors and it is easy to be manufactured, providing also the benefit of low cost.

DOI: 10.17148/IJIREEICE.2025.13603

REFERENCES

[1]. C. Swapna, J.V. Kumar, K.R. Raghavendra and T. Ramanjappa, 'FPGA based Heart Beat Measurement System', International Conference on Information Technology, Electronics and Communications (ICITEC), 2012
[2]. B.K. Rehman, A. Kumar and P. Sharma, 'An FPGA based High Performance Heart Beat Monitoring System', International Journal of Control Theory and Applications, vol. 10, no 18, pp. 245-253, 2017
[3]. Y. Wang, ' A pulse sensor interface design for FPGA based multisensor health monitoring platform', International Journal of Biosensors and Bioelectronics, 5 (1), pp. 23-27, 2019
[4]. K. Meddah, M.K. Talha, M. Bahoura and H. Zairi, ' FPGA-based system for heart rate monitoring', IET Circuits, Devices and Systems, vol. 13, is.6, pp. 771-782, 2019
[5]. J.A.G. Limon, F.M.-Suarez and C.A.-Serrano, 'Implementation of Wavelet-Transform-Based Algorithms in an FPGA for Heart Rate and RT Interval Automatic Measurements in Real Time: Application in a Long-Term Ambulatory Electrocardiogram Monitor', Micromachines, 14, 1748, 2023
[6]. https://www.instructables.com/ECG-Monitoring-System-by-Using-Arduino-or-AD8232/
[7]. https://en.wikipedia.org/wiki/Electrocardiography

APPENDIX

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;
use ieee.std_logic_signed.ALL;
use ieee.std_logic_unsigned.ALL;

entity DE10_Lite_ADC_sensors is
generic(ClockFrequencyHz : integer:=50000000);

port
(
rst : in std_logic;
nRst    : in std_logic;
Seconds : inout integer;
Ticks : inout integer;
led1: out std_logic;
led2: out std_logic;
led3: out std_logic;
led4: out std_logic;
led5: out std_logic;
led6: out std_logic;
led7: out std_logic;
led8: out std_logic;
led9: out std_logic;
led10: out std_logic;
led_blue: buffer std_logic; --vradycardia
led_red: buffer std_logic;--tachycardia
led_yellow: buffer std_logic;--normal
led_blue_out: out std_logic;--vradycardia
led_red_out: out std_logic;--tachycardia
led_yellow_out: out std_logic;--normal

buzzer:out std_logic; --rings on tachycardia or vradycardia
Vr: buffer integer;
pulses_per_minute: buffer integer;

d2bbuf :buffer integer range 0 to 9;
d1bbuf :buffer integer range 0 to 9;
d0bbuf :buffer integer range 0 to 9;
SW0 : in std_logic;
```

IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

Impact Factor 8.414 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 6, June 2025

DOI: 10.17148/IJIREEICE.2025.13603

```
red_led_alarm : out std_logic; --hypercalcemia, hyperkalemia, hypocalcemia
yellow_led_alarm: out std_logic;--coronary ischemia
white_led_alarm: out std_logic;--bundle-branch block
blue_led_alarm: out std_logic;--AV node block
green_led_no_alarm: out std_logic;-- none of the above problems

red_led_alarm_buff : buffer std_logic;-- hypercalcemia, hyperkalemia, hypocalcemia
yellow_led_alarm_buff: buffer std_logic;-- coronary ischemia
white_led_alarm_buff: buffer std_logic;-- bundle-branch block
blue_led_alarm_buff: buffer std_logic;-- AV node block
green_led_no_alarm_buff: buffer std_logic;-- none of the above problems

-- Clocks
ADC_CLK_10: in std_logic;
MAX10_CLK1_50: in std_logic;
MAX10_CLK2_50: in std_logic;
-- KEYs
KEY: in std_logic_vector(1 downto 0);

-- HEX
HEX0: out std_logic_vector(7 downto 0);
HEX1: out std_logic_vector(7 downto 0);
HEX2: out std_logic_vector(7 downto 0);
HEX3: out std_logic_vector(7 downto 0);
HEX4: out std_logic_vector(7 downto 0);
HEX5: out std_logic_vector(7 downto 0);

ARDUINO_IO: inout std_logic_vector(15 downto 0);
ARDUINO_RESET_N: inout std_logic);

-- GPIO
--GPIO: inout std_logic_vector(35 downto 0));
end entity;
architecture DE10_Lite_ADC_sensors_Arch of DE10_Lite_ADC_sensors is

-- Analog to Digital Converter IP core
component myADC is
port
(
clk_clk: in std_logic := 'X';
modular_adc_0_command_valid: in std_logic := 'X';
modular_adc_0_command_channel: in std_logic_vector(4 downto 0) := (others => 'X');
modular_adc_0_command_startofpacket: in std_logic := 'X';
modular_adc_0_command_endofpacket: in std_logic := 'X';
modular_adc_0_command_ready: out std_logic;
modular_adc_0_response_valid: out std_logic;
modular_adc_0_response_channel: out std_logic_vector(4 downto 0);
modular_adc_0_response_data: out std_logic_vector(11 downto 0);
modular_adc_0_response_startofpacket: out std_logic;
modular_adc_0_response_endofpacket: out std_logic;
reset_reset_n: in std_logic
);
end component myADC;
signal modular_adc_0_command_valid: std_logic;
signal modular_adc_0_command_channel: std_logic_vector(4 downto 0);
signal modular_adc_0_command_startofpacket: std_logic;
signal modular_adc_0_command_endofpacket: std_logic;
signal modular_adc_0_command_ready: std_logic;
```

IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

Impact Factor 8.414 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 6, June 2025

DOI: 10.17148/IJIREEICE.2025.13603

```
signal modular_adc_0_response_valid: std_logic;
signal modular_adc_0_response_channel: std_logic_vector(4 downto 0);
signal modular_adc_0_response_data: std_logic_vector(11 downto 0);
signal modular_adc_0_response_startofpacket: std_logic;
signal modular_adc_0_response_endofpacket: std_logic;
signal clk_clk: std_logic;
signal reset_reset_n: std_logic;
type state_machines is (sm0,sm1, sm2, sm3, sm4);
signal sm: state_machines;
-- signals to store conversion results
signal ADCIN1,ADCIN4, ADCIN3,ADCIN2: std_logic_vector(11 downto 0);
signal AD1,AD4, AD3,AD2: std_logic_vector(11 downto 0);
-- signals for BCD digits
signal digit2b, digit1b, digit0b: std_logic_vector(3 downto 0);
signal digit2c, digit1c, digit0c: std_logic_vector(3 downto 0);
signal digit5, digit4, digit3, digit2, digit1, digit0: std_logic_vector(3 downto 0);

-- signal to determine how fast the
-- 7-seg displays will be updated
signal cnt: integer;
signal state_LED: std_logic;
signal state_Vr: integer;
--signal Ticks : integer;


begin
-- ADC port map
adc1: myADC port map
(
modular_adc_0_command_valid => modular_adc_0_command_valid,
modular_adc_0_command_channel => modular_adc_0_command_channel,
modular_adc_0_command_startofpacket => modular_adc_0_command_startofpacket,
modular_adc_0_command_endofpacket => modular_adc_0_command_endofpacket,
modular_adc_0_command_ready => modular_adc_0_command_ready,
modular_adc_0_response_valid => modular_adc_0_response_valid,
modular_adc_0_response_channel => modular_adc_0_response_channel,
modular_adc_0_response_data => modular_adc_0_response_data,
modular_adc_0_response_startofpacket => modular_adc_0_response_startofpacket,
modular_adc_0_response_endofpacket => modular_adc_0_response_endofpacket,
clk_clk => clk_clk,
reset_reset_n => reset_reset_n
);
clk_clk <= MAX10_CLK1_50;
reset_reset_n <= KEY(0);

-- process for reading new samples
p1: process(reset_reset_n, clk_clk)
begin
if reset_reset_n = '0' then
        sm <= sm0;
elsif rising_edge(clk_clk) then
        case sm is
                when sm0 =>
                        sm <= sm1;
        modular_adc_0_command_valid <= '1';
modular_adc_0_command_channel <= "00001";
                when sm1 =>
if modular_adc_0_response_valid = '1' then
```

IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

Impact Factor 8.414 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 6, June 2025

DOI: 10.17148/IJIREEICE.2025.13603

```
modular_adc_0_command_channel <= "00010";
 ADCIN4 <= modular_adc_0_response_data;
                                        sm <= sm2;
                                    end if;
                            when sm2 =>
if modular_adc_0_response_valid = '1' then
modular_adc_0_command_channel <= "00001";
modular_adc_0_command_channel <= "00011";
ADCIN1 <= modular_adc_0_response_data;
                                        sm <= sm1;
                                        sm <= sm3;
                                    end if;
                            when sm3 =>
if modular_adc_0_response_valid = '1' then
modular_adc_0_command_channel <= "00100";
ADCIN2 <= modular_adc_0_response_data;
                                        sm <= sm4;
                                    end if;
                            when sm4 =>
if modular_adc_0_response_valid = '1' then
modular_adc_0_command_channel <= "00001";
ADCIN3 <= modular_adc_0_response_data;
                                        sm <= sm1;
                                    end if;
                            when others =>
                end case;
end if;
end process;

-- process for conversion from binary to BCD (analog input voltage)
p3: process(AD2,d2bbuf,d1bbuf,d0bbuf)
variable vin: integer;
variable d2, d1, d0: integer;
begin
vin := to_integer(signed(std_logic_vector(to_signed(to_integer(signed(AD2)) * 500,
32))(31 downto 12)));
d2 := vin / 100;
d1 := vin mod 100 / 10;
d0 := ((vin mod 100) mod 10);
digit2b <= std_logic_vector(to_signed(d2, 4));
digit1b <= std_logic_vector(to_signed(d1, 4));
digit0b <= std_logic_vector(to_signed(d0, 4));
d2bbuf<= d2;
d1bbuf<= d1;
d0bbuf<= d0;
end process;

state_Vr<= (d2bbuf*100)+(d1bbuf*10)+(d0bbuf);
Vr<= state_Vr;

-- determine how fast the 7-seg displays will be updated
p4: process(reset_reset_n, clk_clk)
begin
if reset_reset_n = '0' then
cnt <= 0;
elsif rising_edge(clk_clk) then
if cnt < 20_000_000 then
cnt <= cnt + 1;
```

IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

Impact Factor 8.414 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 6, June 2025

DOI: 10.17148/IJIREEICE.2025.13603

```
else
cnt <= 0;
AD1 <= ADCIN1;
AD2 <= ADCIN2;
AD3 <= ADCIN3;
AD4 <= ADCIN4;
end if;
end if;
end process;

--time
process(MAX10_CLK1_50) is
   begin
      if rising_edge(MAX10_CLK1_50) then

         ------ If the negative reset signal is active
         if nRst = '0' then
            Ticks   <= 0;
            Seconds <= 0;

         else

            -- True once every second
            if Ticks = ClockFrequencyHz - 1 then
               Ticks <= 0;

                  Seconds <= Seconds + 1;

            else
               Ticks <= Ticks + 1;
            end if;

         end if;
      end if;
   end process;

process (Vr,Ticks,MAX10_CLK1_50)

variable x1: integer;
variable x11:integer;
variable x12: integer;
variable x2: integer;
variable x3: integer;
variable x4: integer;
variable x5: integer;
variable x7: integer;
variable x6: integer;
variable x61: integer;
variable x62: integer;

begin

IF  Ticks<=750000000 THEN--50000000 Ticks= 1sec
IF Vr >=10 AND Vr<= 25 THEN
x1:= Ticks; --P wave peak
end if;

IF Vr>=70  THEN --AND Ticks>x1
```

IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

Impact Factor 8.414 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 6, June 2025

DOI: 10.17148/IJIREEICE.2025.13603

```
x4:= Ticks;  --R wave 1st peak
end if;

IF Vr>=10 AND Vr<=50 THEN --AND Ticks>x1 AND Ticks>x4
x6:= Ticks; --T wave peak
end if;
x11:= x1- (5/100)*50000000;-- taking into account bibliograrhic references that P wave duration is 0.10sec
x12:= x1 +(5/100)*50000000;
x61:= x6 -(75/1000)*50000000; -- taking into account bibliograrhic references that T wave duration is 0.15sec
x62:= x6 +(75/1000)*50000000;
x2:= x12 +(1/10)*50000000; -- taking into account bibliograrhic references that PR segment duration is 0.1sec
IF Vr>=-40 AND Vr<=-25  THEN --AND Ticks>x1
x3:= Ticks; --Q wave peak
end if;
IF Vr>-80 AND Vr<-75  THEN --AND Ticks>x1 AND Ticks>x4
x5:= Ticks; -- S wave peak
end if;
x7:= x5 +(33/1000)*50000000; -- taking into account bibliograrhic references that QRS duration is 0.1sec

IF (x62 - x2)> (3/10)*50000000 OR (x62 - x2) < (3/10)*50000000 THEN --short or long QT interval
red_led_alarm_buff<= '1';
end if;
IF (Ticks>= x61 AND Ticks<= x62) AND (Vr<=0) THEN--flat or inverted T waves
yellow_led_alarm_buff<= '1';
end if;

IF (x7 - x2)> (1/10)*50000000 THEN-- increased QRS interval
white_led_alarm_buff<= '1';
end if;

IF (x2 - x11)> (2/10)*50000000 THEN-- increased PR interval
blue_led_alarm_buff<= '1';
end if;
end if;
pulses_per_minute<= ((1/((2*x4)/50000000))*60);
end process;

process(pulses_per_minute,digit2c,digit1c,digit0c)
variable d2, d1, d0: integer;
begin
d2:= pulses_per_minute/ 100;
d1:= pulses_per_minute mod 100/ 10;
d0:= ((pulses_per_minute mod 100) mod 10);
digit2c <= std_logic_vector(to_unsigned(d2, 4));
digit1c <= std_logic_vector(to_unsigned(d1, 4));
digit0c <= std_logic_vector(to_unsigned(d0, 4));
end process;

--critical pulses_per_minute value exceeded buzzer sounds
Process(MAX10_CLK1_50,pulses_per_minute)
variable i : integer := 0;
BEGIN
IF (pulses_per_minute>100) OR (pulses_per_minute<60) THEN
if MAX10_CLK1_50'event and MAX10_CLK1_50 = '1' then
if i <= 50000000 then
i := i + 1;
buzzer <= '1';
elsif i > 50000000 and i < 100000000 then
```

IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526
International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering
Impact Factor 8.414 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 6, June 2025
DOI: 10.17148/IJIREEICE.2025.13603

```
i := i + 1;
buzzer <= '0';
elsif i = 100000000 then
i := 0;
end if;
end if;
end if;
end process;

--critical pulses_per minute value 100 exceeded, or less than 60 board LEDs and external red led lights up
process(state_LED,led_red,pulses_per_minute, led_blue, led_yellow)
begin
  IF pulses_per_minute>100 THEN
    led_red <= '1';
            state_LED <= '1';
    led_blue <= '0';
    led_yellow <= '0';

  elsif pulses_per_minute<=100 and pulses_per_minute>=60 THEN
    led_red <= '0';
            led_blue <='0';
            led_yellow <='1';
            state_LED <= '0';
  elsif pulses_per_minute<60 THEN
    led_blue <= '1';
            state_LED <= '1';
            led_red <= '0';
            led_yellow <= '0';
  end if;
end process;
led_red_out<= led_red;
led_blue_out<= led_blue;
led_yellow_out<= led_yellow;
led1 <= state_LED;
led2 <= state_LED;
led3 <= state_LED;
led4 <= state_LED;
led5 <= state_LED;
led6 <= state_LED;
led7 <= state_LED;
led8 <= state_LED;
led9 <= state_LED;
led10 <= state_LED;

--alarm level for several heart problems extracted from the electrocardiogram
process(red_led_alarm_buff,yellow_led_alarm_buff,white_led_alarm_buff,
blue_led_alarm_buff,green_led_no_alarm_buff) -- none of the above problems
begin
IF ((red_led_alarm_buff ='0') AND (yellow_led_alarm_buff ='0')
AND (white_led_alarm_buff ='0') AND (blue_led_alarm_buff ='0')) THEN
green_led_no_alarm_buff <= '1';
end if;
end process;

red_led_alarm <= red_led_alarm_buff;
yellow_led_alarm <= yellow_led_alarm_buff;
white_led_alarm <= white_led_alarm_buff;
blue_led_alarm <= blue_led_alarm_buff;
```

IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

Impact Factor 8.414 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 6, June 2025

DOI: 10.17148/IJIREEICE.2025.13603

```
green_led_no_alarm <= green_led_no_alarm_buff;

process(digit2c, digit1c, digit0c, SW0,digit2b,digit1b,digit0b)
begin
  IF SW0='0' THEN
  digit2 <= digit2b;--first digit of ECG input voltage (units)
  digit1 <= digit1b;--second digit of ECG input voltage (10^-1)
  digit0 <= digit0b;--third digit of ECG input voltage (10^-2)
  digit5 <= digit2c;--first digit of pulses/min
  digit4 <= digit1c;--second digit of pulses/min
  digit3 <= digit0c;--third digit of pulses/min
  end if;
end process;

  WITH digit5 SELECT
HEX5 <= "11000000" WHEN "0000", -- display 0
"11111001" WHEN "0001", -- display 1
"10100100" WHEN "0010", -- display 2
"10110000" WHEN "0011", -- display 3
"10011001" WHEN "0100", -- display 4
"10010010" WHEN "0101", -- display 5
"10000011" WHEN "0110", -- display 6
"11111000" WHEN "0111", -- display 7
"10000000" WHEN "1000", -- display 8
"10011000" WHEN "1001", -- display 9
"11111111" WHEN OTHERS; -- blank display

WITH digit4 SELECT
HEX4 <= "11000000" WHEN "0000", -- display 0
"11111001" WHEN "0001", -- display 1
"10100100" WHEN "0010", -- display 2
"10110000" WHEN "0011", -- display 3
"10011001" WHEN "0100", -- display 4
"10010010" WHEN "0101", -- display 5
"10000011" WHEN "0110", -- display 6
"11111000" WHEN "0111", -- display 7
"10000000" WHEN "1000", -- display 8
"10011000" WHEN "1001", -- display 9
"11111111" WHEN OTHERS; -- blank display

WITH digit3 SELECT
HEX3 <=
"11000000" WHEN "0000", -- display 0
"11111001" WHEN "0001", -- display 1
"10100100" WHEN "0010", -- display 2
"10110000" WHEN "0011", -- display 3
"10011001" WHEN "0100", -- display 4
"10010010" WHEN "0101", -- display 5
"10000011" WHEN "0110", -- display 6
"11111000" WHEN "0111", -- display 7
"10000000" WHEN "1000", -- display 8
"10011000" WHEN "1001", -- display 9
"11111111" WHEN OTHERS; -- blank display

WITH digit2 SELECT
HEX2 <= "01000000" WHEN "0000", -- display 0
"01111001" WHEN "0001", -- display 1
"00100100" WHEN "0010", -- display 2
```

IJIREEICE ISSN (O) 2321-2004, ISSN (P) 2321-5526

International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering

Impact Factor 8.414 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 6, June 2025

DOI: 10.17148/IJIREEICE.2025.13603

```
"00110000" WHEN "0011", -- display 3
"00011001" WHEN "0100", -- display 4
"00010010" WHEN "0101", -- display 5
"00000011" WHEN "0110", -- display 6
"01111000" WHEN "0111", -- display 7
"00000000" WHEN "1000", -- display 8
"00011000" WHEN "1001", -- display 9
"01111111" WHEN OTHERS; -- blank display

WITH digit1 SELECT
HEX1 <= "11000000" WHEN "0000", -- display 0
"11111001" WHEN "0001", -- display 1
"10100100" WHEN "0010", -- display 2
"10110000" WHEN "0011", -- display 3
"10011001" WHEN "0100", -- display 4
"10010010" WHEN "0101", -- display 5
"10000011" WHEN "0110", -- display 6
"11111000" WHEN "0111", -- display 7
"10000000" WHEN "1000", -- display 8
"10011000" WHEN "1001", -- display 9
"11111111" WHEN OTHERS; -- blank display

WITH digit0 SELECT
HEX0 <= "11000000" WHEN "0000", -- display 0
"11111001" WHEN "0001", -- display 1
"10100100" WHEN "0010", -- display 2
"10110000" WHEN "0011", -- display 3
"10011001" WHEN "0100", -- display 4
"10010010" WHEN "0101", -- display 5
"10000011" WHEN "0110", -- display 6
"11111000" WHEN "0111", -- display 7
"10000000" WHEN "1000", -- display 8
"10011000" WHEN "1001", -- display 9
"11111111" WHEN OTHERS; -- blank display

end architecture;
```