IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 6, June 2026

DOI: 10.17148/IARJSET.2026.13607

A simple, low cost, three-octave frequency generating system, using two passive buzzers and thirty-six touch capacitive switches, implemented with FPGA and VHDL, suitable for hands-on practice of music beginners

Dr Evangelos I. Dimitriadis[1], Theodora Dimitriadou[2], Leonidas Dimitriadis[3], Stergios Michalakopoulos[4]

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

MSc English Language and Literature, Aristotle University of Thessaloniki, Greece[2]

Undergraduate student, Department of Information and Electronic Engineering, International Hellenic University, 57400, Sindos Thessaloniki, Greece[3]

Informatics and Electronic Engineering, Serres[4]

Abstract: A frequency generating system, capable of producing 36 different sound frequencies, corresponding to tones and semitones of 3rd, 4th and 5th music octaves, is manufactured, programmed and presented here. The system uses two passive buzzers from which the left one is programmed to produce 3rd octave sound frequencies, while the right buzzer produces middle 4th and 5th octave sound frequencies. Our system also uses two external LEDs related to corresponding buzzers and each LED lights up if signal is sent to its relative buzzer. An additional information is given to system user by the VHDL program, showing in FPGA board's seven-segment displays, letters FA or SOL corresponding to FA or SOL keys, if left or right buzzer, respectively, is activated. Both FA and SOL are shown in case that user plays a chord from 3rd – 4th or 3rd – 5th octaves. The system is implemented using DE10-Lite FPGA board and 36 touch-capacitive switches, acting as inputs to the board and obviously related to corresponding notes frequencies. Our system has low manufacturing cost and it is ideal for use in hands-on practicing method of music beginners or in any other application where specific acoustic music notes sounds are needed. The VHDL program used here, also provides the ability of expanding our system by incorporating 36 instead of 2 different output buzzers, each one corresponding to specific tone or semitone notes of the three octaves mentioned above. Our system's capabilities can also be expanded due to the flexibility of the VHDL code, by programming it to generate frequency sounds corresponding to quarter tones.

Keywords: Octaves, music notes, acoustic frequency generator FPGA, VHDL, Buzzer, LEDs.

INTRODUCTION

It is well known that FPGAs have attracted attention of a great number of researchers in the recent years, for industrial as well as for a variety of other applications. [1-15] 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.

Although a lot of work has been done concerning implementation of FPGAs in a variety of applications as mentioned above, there are few works [16-20] dealing with FPGA assisted, music notes frequency systems. The above works: a)Analyze analog audio signals from a digital piano using an FPGA-based Fast Fourier Transform (FFT) for real-time frequency analysis, b)Focus on detecting multiple concurrent notes (chords) in real-time, c)Implement a convolutional spiking network on an FPGA to classify musical notes, d)Describe implementing IIR loss filters on a Cyclone II FPGA to simulate stringed instrument sounds and e)Cover designing a basic piano keyboard, including calculating bit counters for 13 notes in a low octave.

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 6, June 2026

DOI: 10.17148/IARJSET.2026.13607

All of the above works do not solve the problem of manufacturing and programming an FPGA-based, simple, portable, easy to use and low cost system capable of generating a wide range of specific music notes acoustic sounds, in the range of 3 octaves by using 36 capacitive-touch switches and two buzzers. A system which is ideal for use from music beginners and especially little kids.

We present here a system that can generate all music tones and semitones in the range of 3rd, middle 4th and 5th octaves, by simply touching corresponding capacitive switches.

Our system also provides indication to beginner in FPGA's seven-segment displays, showing FA or SOL key depending on the octave used, simultaneously with external LED indication giving proof about the buzzer used upon touching the specific switch.

Another benefit of our system is that it can work with a variety of touch switches and its VHDL program provides the advantage of expanding the system by using 36 buzzers instead of 2, as well as modifying the program in order to generate sounds of all music octaves including tones, semitones and quarter tones.

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. Subsequent Figure 3(a,b,c) presents the implemented notes frequency generating system of this work.

It is obvious from the above figures that our system, except from DE10-Lite FPGA board, contains also some basic circuit parts. It uses two passive buzzer units with their corresponding external red LEDs. LEDs are protected from overloading and subsequent destruction, by 330 ohm series resistance. Each one of the buzzer units uses an NPN transistor with 1K resistance connected at the base pin, for amplification of the FPGA's output frequency signal. Our system also uses 36 touch-capacitive switches connected in different FPGA's GPIO pins, acting as the main input and determine specific notes that user wants to produce. The above switches, as shown in Figure 3, are arranged in the order of the piano keys, which means that from left to right, switches 1 to 12 correspond to 3rd piano octave including all tones and semitones, switches 13 to 24 correspond to middle 4th piano octave and switches 25 to 36 correspond to 5th piano octave.

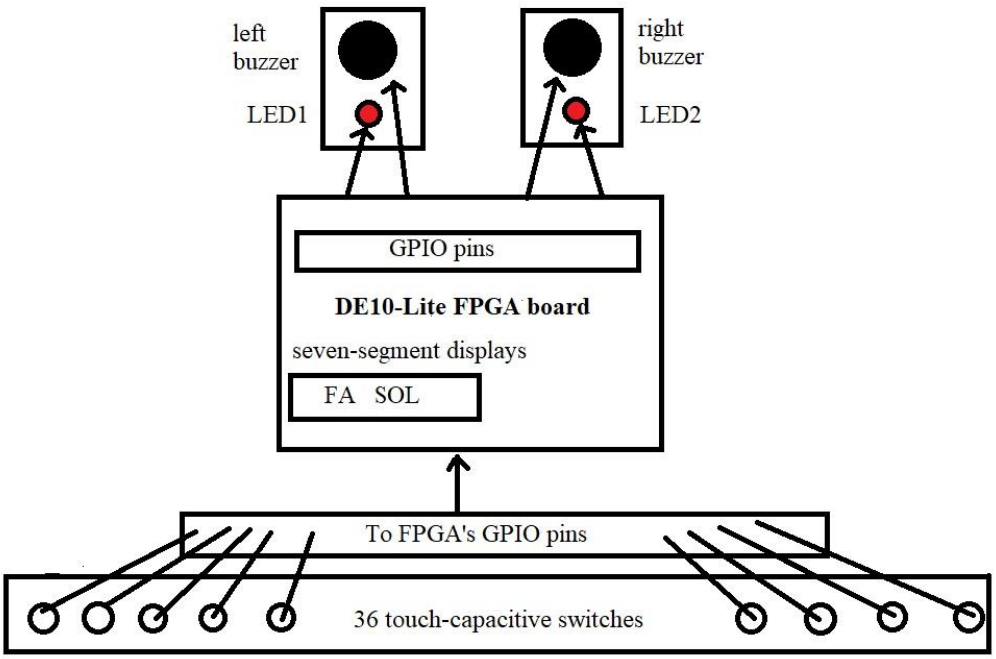


Figure 1: Device overview and operational units of our system.

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 6, June 2026

DOI: 10.17148/IARJSET.2026.13607

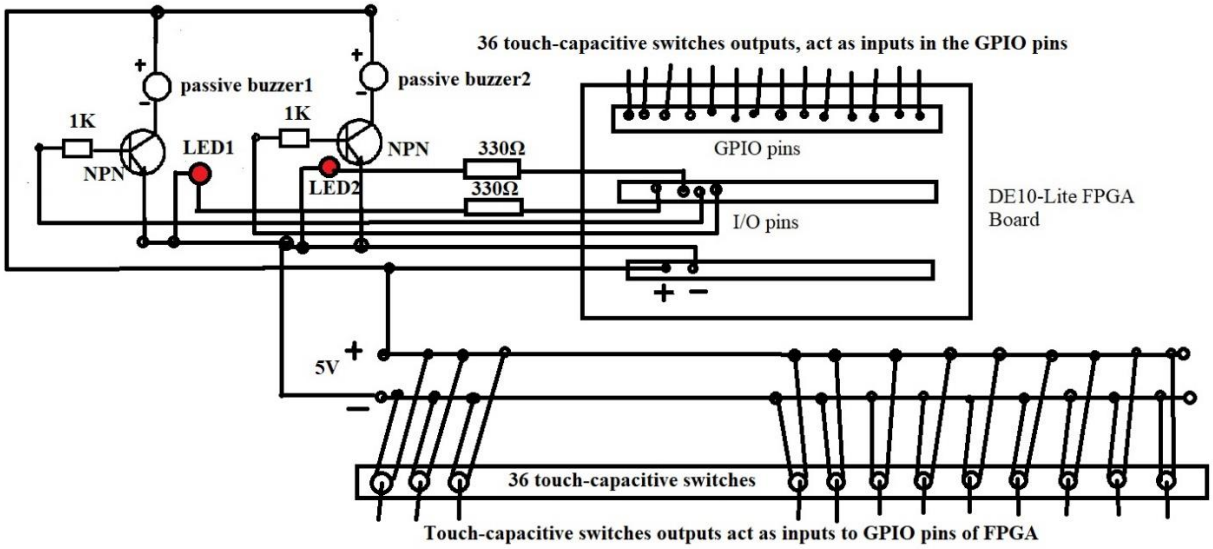
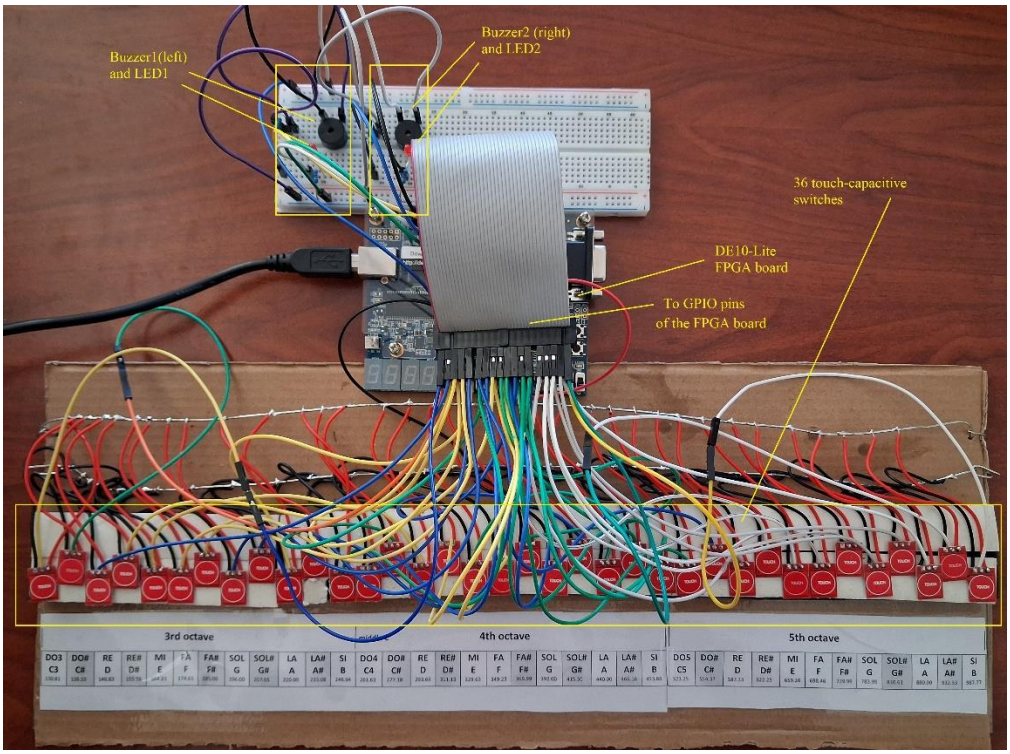


Figure 2: Circuit diagram of our system



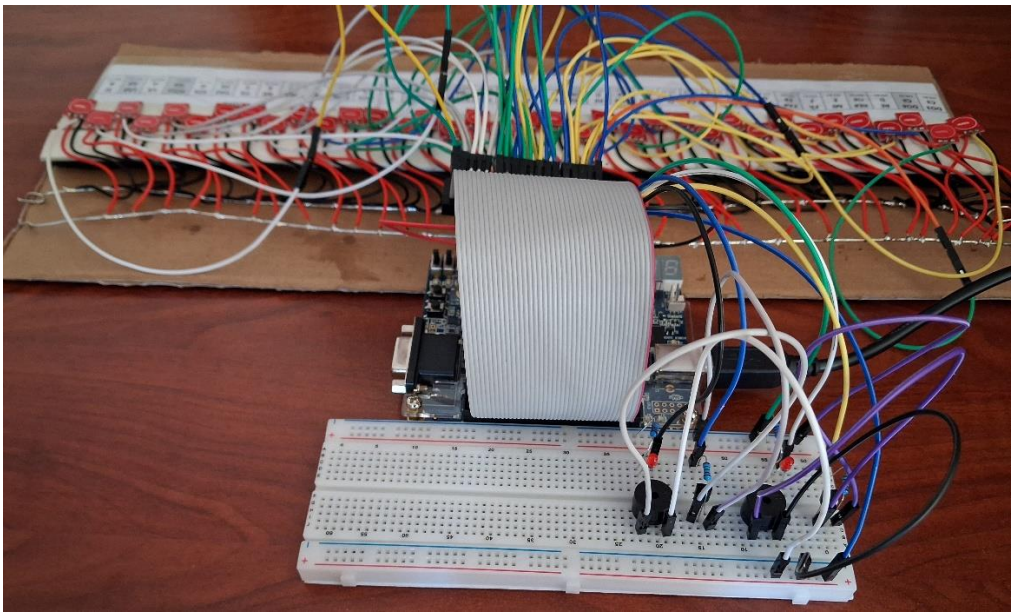
a)

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

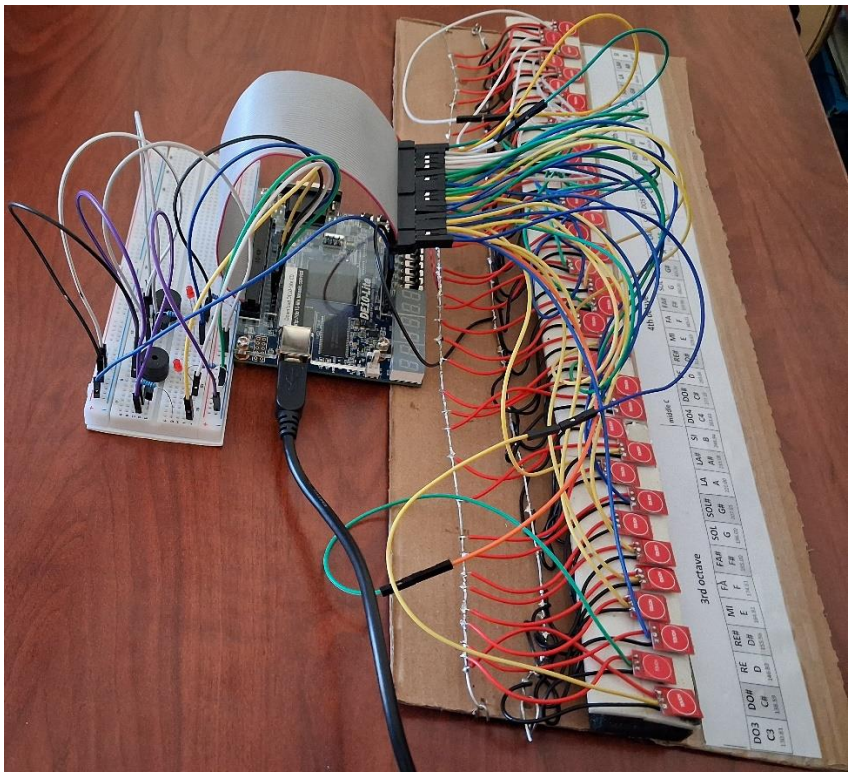
International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 6, June 2026

DOI: 10.17148/IARJSET.2026.13607



b)



c)

Figure 3a), b) and c): The implemented notes frequency generating system of this work.

Our notes frequency generating system starts operating as soon as power supply +5V is applied to the circuits via DE10-Lite appropriate pins and the VHDL program is sent via USB Blaster interface, to FPGA chip.

Input values represented by logic ′1′ from touch-capacitive switches, are send to specific GPIO pins of the FPGA board, whenever a switch is touched by user. Subsequently corresponding signal produced by the VHDL program activates one or both buzzers, depending on the number of switches that are being touched. The system is able to generate one note per buzzer at specific time. This means that if user wants to play music chords he can touch at the same time

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 6, June 2026

DOI: 10.17148/IARJSET.2026.13607

one of 3rd octave switches and simultaneously one switch from 4th or 5th octave. We mentioned above that if we want to expand our system's abilities we can add 36 instead of 2 passive buzzers, in order to be able to generate sounds by simultaneously touching more than one switches from the preferred octaves. However there is a limit to the above expansion due to FPGA's limited number of GPIO or I/O pins.

Upon activation of a buzzer its corresponding external red LED lights up, providing indication about the sounding buzzer. Needless to say that if both buzzers sound then both LEDs are ON. Our system also provides another useful indication to music beginners and especially little kids by displaying the name of the music key FA or SOL in which the generated music note corresponding to specific touched switch, is included. FA is displayed in left seven-segment displays of the board if a switch of 3rd octave is touched, while SOL is displayed in case that 4th or 5th octave switches are being touched. In case that both buzzers are simultaneously activated then both FA and SOL keys appear in board's seven-segment displays.

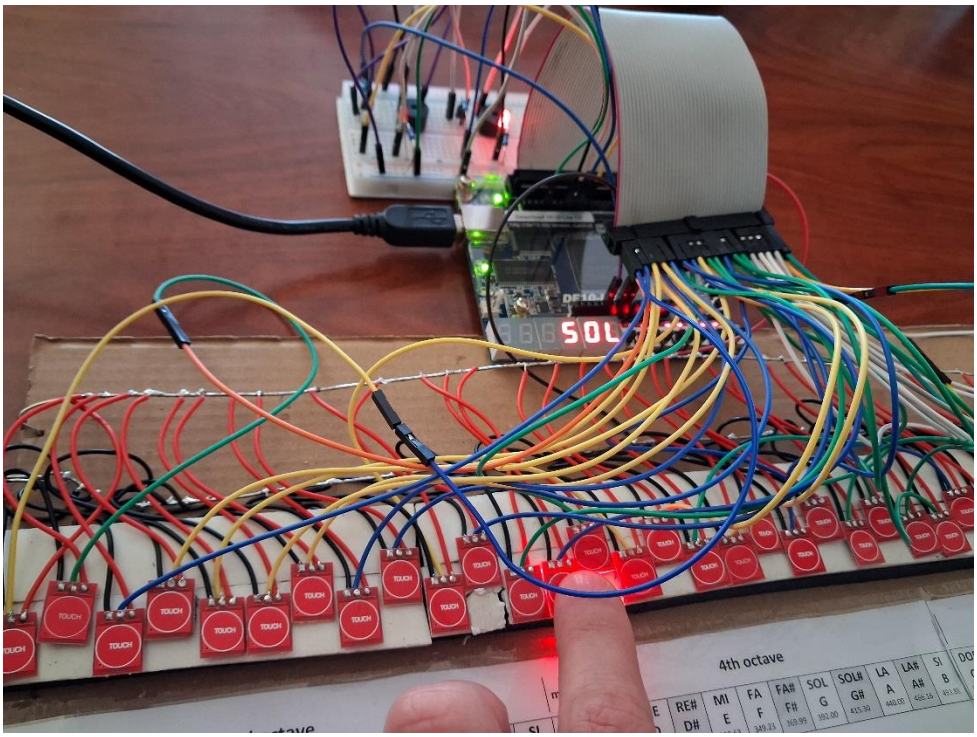


Figure 4: The notes frequency generating system operating. Touch switches 13 to 36 are used and right buzzer with its corresponding external LED are activated. Octaves 4 and 5 are used with simultaneous SOL displaying in board's seven-segment displays

Figure 4 presents our notes frequency generating system in operating condition. We use touch switches 13 to 36, thus right buzzer with its corresponding external LED are activated. Octaves 4 and 5 are used and simultaneously SOL is displayed in board's seven-segment displays.

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 6, June 2026

DOI: 10.17148/IARJSET.2026.13607

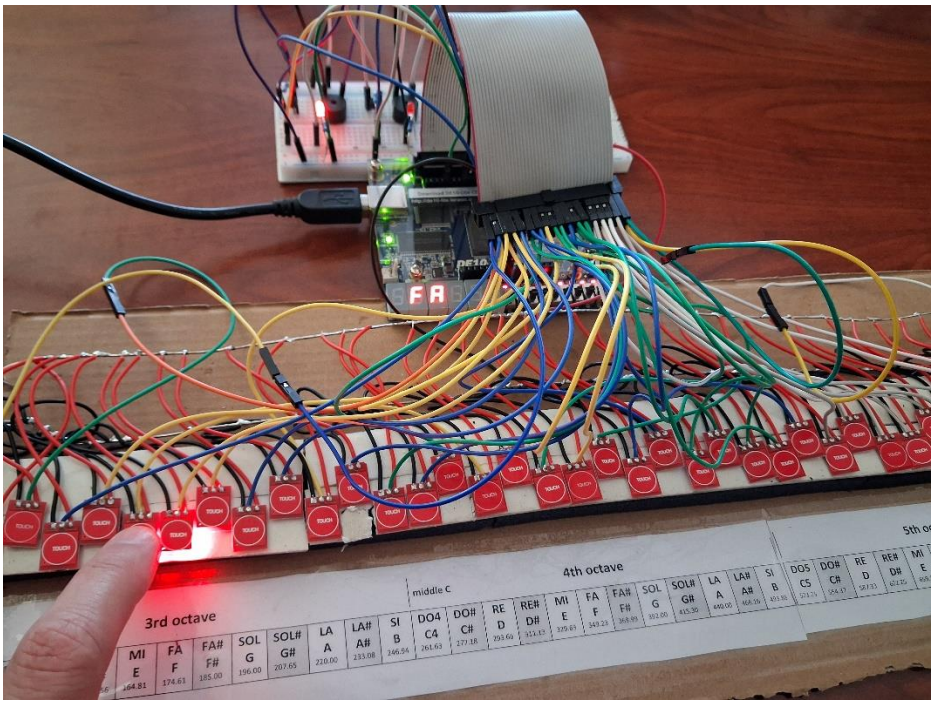
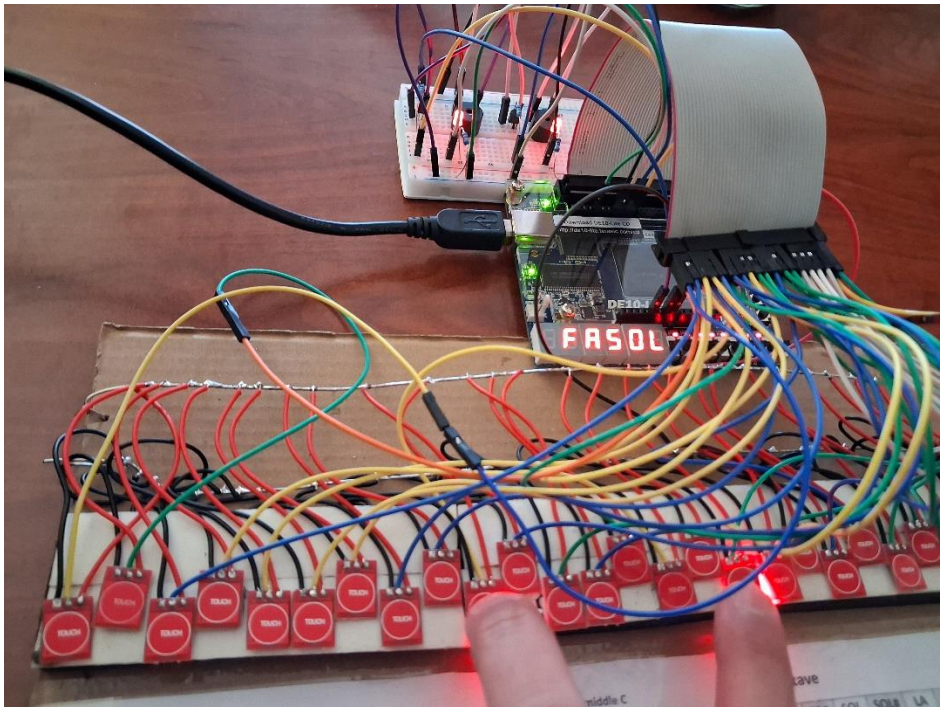


Figure 5: The notes frequency generating system operating. Touch switches 1 to 12 are used and left buzzer with its corresponding external LED are activated. Octave 3 is used with simultaneous FA displaying in board's seven-segment displays

Figure 5 also shows the notes frequency generating system, operating. Now touch-capacitive switches 1 to 12 are used and left buzzer with its corresponding external LED are activated. The generated note frequencies are included in octave 3, thus simultaneously FA is displayed in board's seven-segment displays.



IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 6, June 2026

DOI: 10.17148/IARJSET.2026.13607

Figure 6: The notes frequency generating system operating. Touch switches 1 to 36 are used producing chords of 3 and 4 octaves or 3 and 6 octaves, leading to activation of both left and right buzzers and their corresponding external LEDs. Consequently FA and SOL are simultaneously displayed in board's seven-segment displays

Finally in Figure 6 our notes frequency generating system is also in operating mode, but the difference here is that all touch-capacitive switches from 1 to 36 are used. This produces chords from 3 and 4 octaves or 3 and 6 octaves, leading to activation of both right and left buzzers and their corresponding external LEDs. Consequently FA and SOL are both displayed simultaneously in FPGA's seven-segment displays.

Programing the system

We used Quartus Prime Lite Edition 21.1.1 to create the VHDL programs of our system. A flowchart diagram, presenting main functions of our system is presented in Figure 7, while the APPENDIX contains the whole VHDL program.

It is clear that the system operates six basic functions. All of them use processes in VHDL programming language. These processes are running as long as the system is ON. All external LEDs and buzzers, are programmed to work as logic outputs, thus they are at the ON state if they receive binary '1' as logic output.

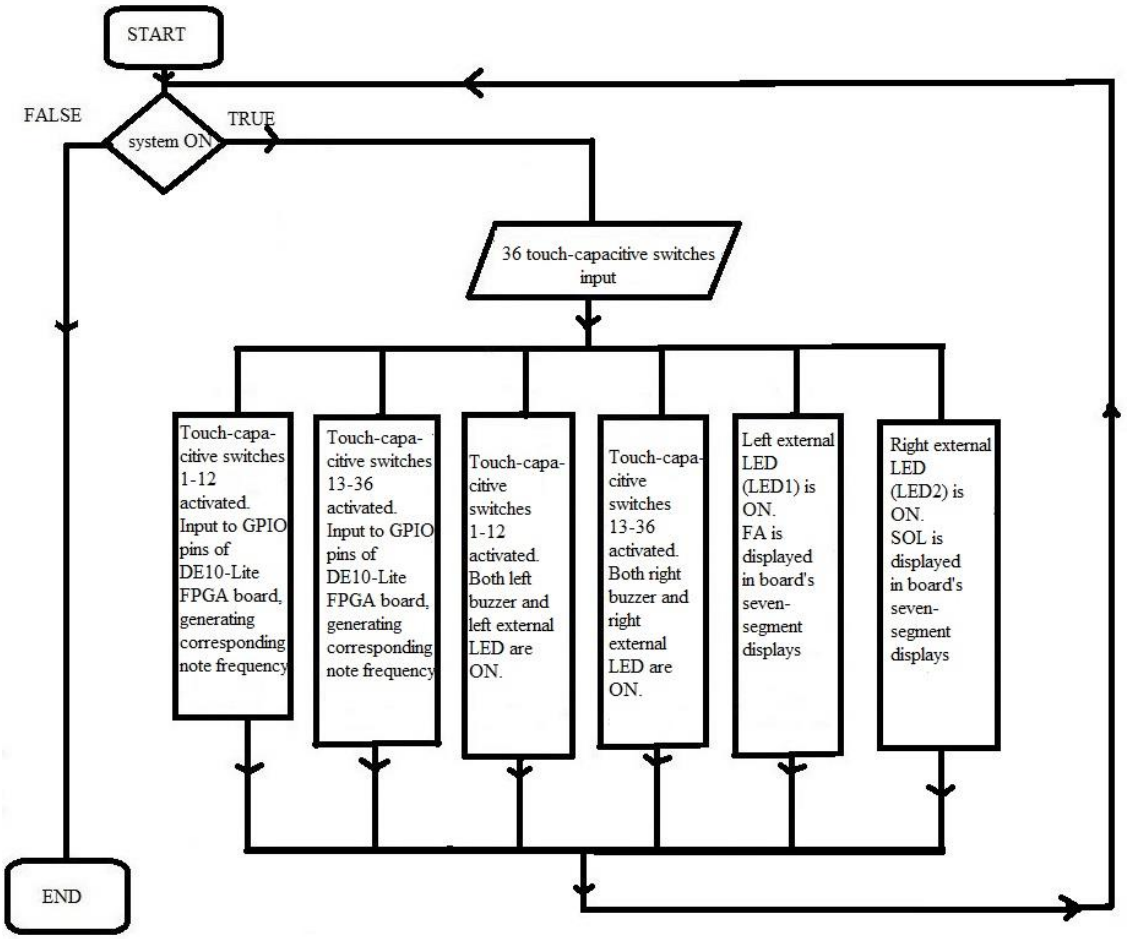


Figure 7: Flowchart diagram, presenting main functions-processes of our system.

First and second functions of Figure 7 play definitive role in system operation. The first process represents 12 simpler processes, each of them being responsible for generating the corresponding tone or semitone signal, depending on the specific touch-capacitive switch that it is touched by user. The above 12 simpler processes constitute 3rd piano octave. Similarly second function represents 24 simpler processes, each of them being responsible for generating the corresponding tone or semitone signal, depending on the specific touch-capacitive switch being touched. These 24 simpler processes constitute 4th and 5th piano octaves.

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 6, June 2026

DOI: 10.17148/IARJSET.2026.13607

The main programming idea in total and similar 36 simpler processes is to set a COUNT LIMIT which determines the frequency that we want to generate for each tone or semitone of 3rd, 4th and 5th octaves. This happens because unlike to active buzzers, the passive buzzers used here, require an external oscillating signal (PWM) to sound, allowing them to produce different tones. An FPGA counter in the VHDL program, counts down from a set value toggling the output pin connected to buzzer, at specific intervals to create the desired frequency. If user wants to produce a note with frequency f (Hz) the counter is configured to toggle every

$\frac{50\ MHz}{2f}$ clock cycles, which is the value of the COUNT LIMIT mentioned above. The clock of DE10-Lite FPGA board works at 50 MHz.

Tables 1, 2 and 3 shown below, present notes and their corresponding frequency and count limit for 3rd, 4th and 5th octaves, respectively, used in 36 simple processes each one of them producing the appropriate output for the passive buzzers of this work and corresponding to a different touch-capacitive switch.

Table1: 3rd octave

Note	**DO3 C3**	**DO# C#**	**RE D**	**RE# D#**	**MI E**	**FA F**	**FA# F#**	**SOL G**	**SOL# G#**	**LA A**	**LA# A#**	**SI B**
Frequency (Hz)	130.81	138.59	146.83	155.56	164.81	174.61	185.00	196.00	207.65	220.00	233.08	246.94
Count Limit 50MHz/ (2*F) No decimals	191116	180388	170264	160709	151689	143176	135135	127551	120394	113636	107259	101239

Table1: Notes, frequencies and corresponding count limits of 3rd octave.

Table2: 4th octave

Note	**DO4 C4**	**DO# C#**	**RE D**	**RE# D#**	**MI E**	**FA F**	**FA# F#**	**SOL G**	**SOL# G#**	**LA A**	**LA# A#**	**SI B**
Frequency (Hz)	261.63	277.18	293.66	311.13	329.63	349.23	369.99	392.00	415.30	440.00	466.16	493.88
Count Limit 50MHz/ (2*F) No decimals	95554	90194	85132	80352	75842	71586	67569	63775	60197	56818	53629	50619

Table2: Notes, frequencies and corresponding count limits of 4th octave.

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 6, June 2026

DOI: 10.17148/IARJSET.2026.13607

Table3: 5th octave

Note	DO5 C5	DO# C#	RE D	RE# D#	MI E	FA F	FA# F#	SOL G	SOL# G#	LA A	LA# A#	SI B
Frequency (Hz)	523.25	554.37	587.33	622.25	659.26	698.46	739.99	783.99	830.61	880.00	932.33	987.77
Count Limit 50MHz/ (2*F) No decimals	47778	45096	42565	40176	37291	35793	33784	31888	30098	28409	26814	25309

Table3: Notes, frequencies and corresponding count limits of 5th octave.

We also observe in Figure7 functions-processes 3 and 4. These processes have a similar operation which is activating the appropriate buzzer and sending it the corresponding signal generated in processes 1 and 2, upon touching one of the 36 switches. Process 3 is activated by using 1-12 switches and sends a signal to left buzzer with simultaneous lighting of left red, external LED (LED1). Process 4 is activated by using 13-36 switches and sends a signal to right buzzer with simultaneous lighting of right red, external LED (LED2).

Finally functions-processes 5 and 6 are checking whether LED1 or LED2, respectively, are activated in order to display the appropriate word in board's seven-segment displays. In process 5 if LED1 is ON this means that we use switches corresponding to 3rd piano octave, thus we are in the range of FA key and FA is displayed. Similarly in process 6 if LED2 is ON this means that we use switches corresponding to 4th or 5th piano octaves, thus we are in the range of SOL key and SOL is displayed.

Application of the system

Hands-on learning is an active educational method where you learn by doing rather than by just reading or listening. By physically engaging with materials, you transform abstract concepts into tangible, real-world experiences. This approach is proven to significantly improve your memory, retention, and critical problem-solving skills [21-22]. This method works because if a human actively uses his physical senses (touch, sight, etc.), helps him to create stronger neural connections and sensory memories. In fact you shift from being a passive observer to an active participant, optimizing how your brain processes new information. Finally hands-on learning bridges the gap between theoretical rules and how those concepts actually operate in practice.

Taking into account all the above we propose the application of our system to music beginners of all ages but especially little kids. It is certain that kids will find our system suitable for gamified hands-on learning. It will help children become familiar with musical notes, tones, semitones, tone duration, octaves and everything a music beginner would like to learn. It is obvious from our manufactured system that small touch-capacitive switches implemented in our work, are ideal for little kids having small fingers and also there is no need to exert power on switches, but use them with a simple touch. Additionally teacher could help children learn the names of music notes as they touch the switches, by asking them to pronounce a simple word that comes to mind and begins with the letters of the note they are playing at that particular moment. This will help them enrich their vocabulary at the same time.

Needless to say that except from the application in music beginners, our system can be useful in every case that specific note frequencies are need to be generated, for example in acoustic frequencies recognition in security systems or medical equipment for testing human ear efficiency.

Finally we present the famous song "Memories" played with our manufactured system: https://www.youtube.com/watch?v=mZ8TSxOSmnU

Conclusions

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 6, June 2026

DOI: 10.17148/IARJSET.2026.13607

An FPGA-based, music notes, frequency generating system, is presented in this work. Our system is able to produce 36 different frequency signals, which correspond to all tones and semitones of 3rd, middle 4th and 5th music octaves. It uses two passive buzzers for playing the notes which FPGA is sending as output, depending on which of 36 touch-capacitive switches is being touched by user. Both buzzers are equipped with an external LED indicating whether corresponding buzzer is ON with simultaneous lighting of LED. Left buzzer is activated if switches 1 to 12 (3rd octave) are touched, while right buzzer is ON if switches 13 to 24 (4th octave) or 25 to 36 (5th octave) are touched. Additional indication is shown to user, by displaying in FPGA's seven-segment displays FA (key) if left buzzer produces sounds and SOL (key) if right buzzer produces sounds. Our system is easy to use and low cost to manufacture. System's VHDL program provides the ability for system to be expanded by adding more buzzers for better behavior when playing chords and also programming it to produce quarter tones. Our manufactured note frequencies generating system is ideal for hands-on learning use from music beginners and especially kids and also can be applicable in medical or security sound frequencies generating systems.

REFERENCES

[1]. M. Yussup, M.M. Ibrahim, L. Lombigit, N.A.A. Rahman and M.R.M. Zin, "Implementation of data acquisition interface using on-board field-programmable gate array (FPGA) universal serial bus (USB) link", Advanc. in Nuclear Research and Energy Development, AIP Conf. Proc. 1584, 69-72 (2014), doi: 10.1063/1.4866106

A. Gujar, International Journal of Computer Science and Information Technologies, "Image Encryption using AES Algorithm based on FPGA", vol 5, (5), 2014, 6853-6859

[2]. S. Singh, A.K. Saini, R. Saini, I.J. Image, Graphics and Signal Processing, "Interfacing the Analog Camera with FPGA Board for Real-time Video Acquisition" 2014, 4, 32-38, DOI: 10.5815/ijigsp.2014.04.04

[3]. S. Muthukrishnan and R. Priyadharsini, International Journal of Computer Science and Mobile Computing, "32-Bit RISC and DSP System Design in an FPGA" vol 3, issue 12, Dec. 2014, pg. 361-368

[4]. L. Chen, Y. Chang, L. Yan, IEEE Transactions on Geoscience and Remote Sensing, "On-orbit real-time variational image destriping: FPGA architecture and implementation", 10.1109/tgrs.2022.3140428, 2022, pp. 1-1

[5]. S. Yarlagadda, S. Kaza, A. Tummala, E. Babu, R. Prabhakar, Information Technology in Industry, "The reduction of Crosstalk in VLSI due to parallel bus structure using Data Compression Bus Encoding technique implemented on Artix 7 FPGA Architecture", 10.17762/itii.v9i1.151, 2021, Vol 9 (1), pp. 456-460

[6]. C. Du, Y. Yamaguchi, Electronics, "High-Level Synthesis Design for Stencil Computations on FPGA with High Bandwidth Memory", 2020, 9(8), 1275; https://doi.org/10.3390/electronics9081275

[7]. X. Hao, C. Lin and Q. Wu, Electronics, "A Parallel Timing Synchronization Structure in Real-Time High Transmission Capacity Wireless Communication Systems", 2020, 9(4), 652; https://doi.org/10.3390/electronics9040652

[8]. P. A. Bawiskar, R.K. Agrawal, International Journal of Innovative Research in Science, Engineering and Technology, "FPGA Based Home Security System" vol.4, issue 12, Dec. 2015, p. 12865-12869, DOI: 10.15680/IJIRSET.2015.0412139

[9]. K. Saroch, A. Sharma, IOSR Journal of Electronics and Communication Engineering, "FPGA Based System Login Security Lock Design Using Finite

A. State Machine" vol 5, issue 3, Mar.-Apr. 2013, pp 70-75

[10]. R.S. Parikh, Int. Journal of Engineering Research and Application, "Alarm System Implementation on Field Programmable Gate Array" vol 8, issue 1, Jan. 2018, pp 01-04.

[11]. E. I. Dimitriadis and L. Dimitriadis, "A Simple, Low Cost and Multiple Input Alarm System, Functioning as Finite State Machine (FSM), Using VHDL and FPGAs", Journal of Active and Passive Electronic Devices, vol. 17, pp. 307–315, 2024.

[12]. E. I. Dimitriadis and L. Dimitriadis, "A g-sensor based alarm system, for multiple tilt sensor applications, using VHDL and FPGAs", Journal of Active and Passive Electronic Devices, vol. 18, pp. 119-129, 2024.

[13]. Evangelos I. Dimitriadis, Leonidas Dimitriadis and Aristeidis Grigoriadis, "A touch-sensing system for precise robotic arm control, using force-sensing resistors (FSR), VHDL and FPGAs", International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering, vol. 13, issue 7, pp. 36-50, July 2025.

[14]. Leonidas Dimitriadis and Evangelos I. Dimitriadis, "A Novel System for Monitoring and Controlling Industrial Engine Operation, using Vibration, Magnetic Field, Humidity and Temperature Sensors, Implemented with FPGAs and VHDL", International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering, vol. 14, issue 5, pp. 1-18, May 2026.

[15]. Shafayet M. Anik, D.G. Perera, "Real-Time Piano Note Frequency Detection Using FPGA and FFT Core", arXiv:2509.00589, 2025, https://doi.org/10.48550/arXiv.2509.00589

[16]. Shuyi Chen, Lizi George and Kelly Ran, "Polyphonic Music Transcription on an FPGA", MIT, 2013

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 6, June 2026

DOI: 10.17148/IARJSET.2026.13607

[17]. E. Cerezuela-Escudero, A. Jimenez-Fernandez, R. Paz-Vicente, M. Dominguez-Morales, A. Linares-Barranco, G. Jimenez-Moreno, “Musical notes classification with neuromorphic auditory system using FPGA and a convolutional spiking network”, International Joint Conference on Neural Networks (IJCNN), 2015

[18]. Siti Aisyah et.al.,“FPGA-based sound synthesis by digital waveguide”, 6th International Conference on Modeling, Simulation, and Applied Optimization (ICMSAO), 2015, DOI: 10.1109/ICMSAO.2015.7152258

[19]. M.A. Asyraf Khairuddin, Nabihah Ahmad, Mohd Helmy Abd Wahab, Syed Zulkarnain, Syed Idrus, “Digital Piano Keyboard Design using FPGA Implementation for Beginner”, International Conference on Technology, Engineering and Sciences (ICTES), 917, 2020, doi:10.1088/1757-899X/917/1/012053

[20]. Tom van Eijck, Bert Bredeweg, Joanna Holt, Monique Pijls, Anders Bouwer, Anna Hotze, "Combining hands-on and minds-on learning with interactive diagrams in primary science education", International Journal of Science Education, Volume 47, Issue 18, 2025, p.2413-2433, https://doi.org/10.1080/09500693.2024.2387225

[21]. Nesra Yannier, Scott E. Hudson, Kenneth R. Koedinger, Kathy Hirsh-Pasek, Roberta Michnick Golinkoff, Yuko Munakata, Sabine Doebel, Daniel L.Schwartz, Louis Deslauriers and Sara E. Brownell, "Active learning: “Hands-on” meets “minds-on”", Science, 30 Sep 2021, Vol 374, Issue 6563, pp. 26-30 DOI: 10.1126/science.abj9957

APPENDIX

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;
entity buzzer_note is
   Port ( clk      : in  STD_LOGIC; -- 50MHz input clock
                                    buzzer1   : out STD_LOGIC;  -- Output to passive buzzer1
                                        buzzer2   : out STD_LOGIC; -- Output to passive buzzer2
                                        SW1      : in  STD_LOGIC;
                                        SW2      : in  STD_LOGIC;
                                        SW3      : in  STD_LOGIC;
                                        SW4      : in  STD_LOGIC;
                                        SW5      : in  STD_LOGIC;
                                        SW6      : in  STD_LOGIC;
                                        SW7      : in  STD_LOGIC;
                                        SW8      : in  STD_LOGIC;
                                        SW9      : in  STD_LOGIC;
                                        SW10     : in  STD_LOGIC;
                                        SW11     : in  STD_LOGIC;
                                        SW12     : in  STD_LOGIC;
                                        SW13     : in  STD_LOGIC;
                                        SW14     : in  STD_LOGIC;
                                        SW15     : in  STD_LOGIC;
                                        SW16     : in  STD_LOGIC;
                                        SW17     : in  STD_LOGIC;
                                        SW18     : in  STD_LOGIC;
                                        SW19     : in  STD_LOGIC;
                                        SW20     : in  STD_LOGIC;
                                        SW21     : in  STD_LOGIC;
                                        SW22     : in  STD_LOGIC;
                                        SW23     : in  STD_LOGIC;
                                        SW24     : in  STD_LOGIC;
                                        SW25     : in  STD_LOGIC;
                                        SW26     : in  STD_LOGIC;
                                        SW27     : in  STD_LOGIC;
                                        SW28     : in  STD_LOGIC;
                                        SW29     : in  STD_LOGIC;
                                        SW30     : in  STD_LOGIC;
                                        SW31     : in  STD_LOGIC;
                                        SW32     : in  STD_LOGIC;
                                        SW33     : in  STD_LOGIC;
                                        SW34     : in  STD_LOGIC;
                                        SW35     : in  STD_LOGIC;
                                        SW36     : in  STD_LOGIC;
                                        LED1     : out STD_LOGIC;
                                        LED2     : out STD_LOGIC;
                                        -- HEX
```

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 6, June 2026

DOI: 10.17148/IARJSET.2026.13607

```
HEX0: out std_logic_vector(7 downto 0); -- L
HEX1: out std_logic_vector(7 downto 0); -- O
HEX2: out std_logic_vector(7 downto 0); -- S
HEX3: out std_logic_vector(7 downto 0); -- A
HEX4: out std_logic_vector(7 downto 0) -- F
      );
end buzzer_note;
architecture Behavioral of buzzer_note is
   constant COUNT_LIMIT1 : integer := 191116;
   signal counter1      : integer range 0 to COUNT_LIMIT1 := 0;
            constant COUNT_LIMIT2 : integer := 180388;
   signal counter2      : integer range 0 to COUNT_LIMIT2 := 0;
            constant COUNT_LIMIT3 : integer := 170264;
   signal counter3      : integer range 0 to COUNT_LIMIT3 := 0;
            constant COUNT_LIMIT4 : integer := 160709;
   signal counter4      : integer range 0 to COUNT_LIMIT4 := 0;
            constant COUNT_LIMIT5 : integer := 151689;
   signal counter5      : integer range 0 to COUNT_LIMIT5 := 0;
            constant COUNT_LIMIT6 : integer := 143176;
   signal counter6      : integer range 0 to COUNT_LIMIT6 := 0;
            constant COUNT_LIMIT7 : integer := 135135;
   signal counter7      : integer range 0 to COUNT_LIMIT7 := 0;
            constant COUNT_LIMIT8 : integer := 127551;
   signal counter8      : integer range 0 to COUNT_LIMIT8 := 0;
            constant COUNT_LIMIT9 : integer := 120394;
   signal counter9      : integer range 0 to COUNT_LIMIT9 := 0;
            constant COUNT_LIMIT10 : integer := 113636;
   signal counter10      : integer range 0 to COUNT_LIMIT10 := 0;
            constant COUNT_LIMIT11 : integer := 107259;
   signal counter11      : integer range 0 to COUNT_LIMIT11 := 0;
            constant COUNT_LIMIT12 : integer := 101239;
   signal counter12      : integer range 0 to COUNT_LIMIT12 := 0;
            constant COUNT_LIMIT13 : integer := 95554;
   signal counter13      : integer range 0 to COUNT_LIMIT13 := 0;
            constant COUNT_LIMIT14 : integer := 90194;
   signal counter14      : integer range 0 to COUNT_LIMIT14 := 0;
            constant COUNT_LIMIT15 : integer := 85132;
   signal counter15      : integer range 0 to COUNT_LIMIT15 := 0;
            constant COUNT_LIMIT16 : integer := 80352;
   signal counter16      : integer range 0 to COUNT_LIMIT16 := 0;
            constant COUNT_LIMIT17 : integer := 75842;
   signal counter17      : integer range 0 to COUNT_LIMIT17 := 0;
            constant COUNT_LIMIT18 : integer := 71586;
   signal counter18      : integer range 0 to COUNT_LIMIT18 := 0;
            constant COUNT_LIMIT19 : integer := 67569;
   signal counter19      : integer range 0 to COUNT_LIMIT19 := 0;
            constant COUNT_LIMIT20 : integer := 63775;
   signal counter20      : integer range 0 to COUNT_LIMIT20 := 0;
            constant COUNT_LIMIT21 : integer := 60197;
   signal counter21      : integer range 0 to COUNT_LIMIT21 := 0;
            constant COUNT_LIMIT22 : integer := 56818;
   signal counter22      : integer range 0 to COUNT_LIMIT22 := 0;
            constant COUNT_LIMIT23 : integer := 53629;
   signal counter23      : integer range 0 to COUNT_LIMIT23 := 0;
            constant COUNT_LIMIT24 : integer := 50619;
   signal counter24      : integer range 0 to COUNT_LIMIT24 := 0;
            constant COUNT_LIMIT25 : integer := 47778;
   signal counter25      : integer range 0 to COUNT_LIMIT25 := 0;
            constant COUNT_LIMIT26 : integer := 45096;
   signal counter26      : integer range 0 to COUNT_LIMIT26 := 0;
            constant COUNT_LIMIT27 : integer := 42565;
   signal counter27      : integer range 0 to COUNT_LIMIT27 := 0;
            constant COUNT_LIMIT28 : integer := 40176;
   signal counter28      : integer range 0 to COUNT_LIMIT28 := 0;
            constant COUNT_LIMIT29 : integer := 37291;
   signal counter29      : integer range 0 to COUNT_LIMIT29 := 0;
            constant COUNT_LIMIT30 : integer := 35793;
   signal counter30      : integer range 0 to COUNT_LIMIT30 := 0;
            constant COUNT_LIMIT31 : integer := 33784;
   signal counter31      : integer range 0 to COUNT_LIMIT31 := 0;
            constant COUNT_LIMIT32 : integer := 31888;
   signal counter32      : integer range 0 to COUNT_LIMIT32 := 0;
            constant COUNT_LIMIT33 : integer := 30098;
```

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 6, June 2026

DOI: 10.17148/IARJSET.2026.13607

```
    signal counter33      : integer range 0 to COUNT_LIMIT33 := 0;
              constant COUNT_LIMIT34 : integer := 28409;
    signal counter34      : integer range 0 to COUNT_LIMIT34 := 0;
              constant COUNT_LIMIT35 : integer := 26814;
    signal counter35      : integer range 0 to COUNT_LIMIT35 := 0;
              constant COUNT_LIMIT36 : integer := 25309;
    signal counter36      : integer range 0 to COUNT_LIMIT36 := 0;
              signal temp_buzzer1   : std_logic; --:= '0';
    signal temp_buzzer2   : std_logic; --:= '0';
              signal temp_buzzer_1   : std_logic := '0';
              signal temp_buzzer_2   : std_logic := '0';
              signal temp_buzzer_3   : std_logic := '0';
              signal temp_buzzer_4   : std_logic := '0';
              signal temp_buzzer_5   : std_logic := '0';
              signal temp_buzzer_6   : std_logic := '0';
              signal temp_buzzer_7   : std_logic := '0';
              signal temp_buzzer_8   : std_logic := '0';
              signal temp_buzzer_9   : std_logic := '0';
              signal temp_buzzer_10   : std_logic := '0';
              signal temp_buzzer_11   : std_logic := '0';
              signal temp_buzzer_12   : std_logic := '0';
              signal temp_buzzer_13   : std_logic := '0';
              signal temp_buzzer_14   : std_logic := '0';
              signal temp_buzzer_15   : std_logic := '0';
              signal temp_buzzer_16   : std_logic := '0';
              signal temp_buzzer_17   : std_logic := '0';
              signal temp_buzzer_18   : std_logic := '0';
              signal temp_buzzer_19   : std_logic := '0';
              signal temp_buzzer_20   : std_logic := '0';
              signal temp_buzzer_21   : std_logic := '0';
              signal temp_buzzer_22   : std_logic := '0';
              signal temp_buzzer_23   : std_logic := '0';
              signal temp_buzzer_24   : std_logic := '0';
              signal temp_buzzer_25   : std_logic := '0';
              signal temp_buzzer_26   : std_logic := '0';
              signal temp_buzzer_27   : std_logic := '0';
              signal temp_buzzer_28   : std_logic := '0';
              signal temp_buzzer_29   : std_logic := '0';
              signal temp_buzzer_30   : std_logic := '0';
              signal temp_buzzer_31   : std_logic := '0';
              signal temp_buzzer_32   : std_logic := '0';
              signal temp_buzzer_33   : std_logic := '0';
              signal temp_buzzer_34   : std_logic := '0';
              signal temp_buzzer_35   : std_logic := '0';
              signal temp_buzzer_36   : std_logic := '0';
              signal LED1_sig : std_logic;
              signal LED2_sig : std_logic;
begin
    process(clk, SW1)
    begin
              if SW1= '1' THEN
                                  if rising_edge(clk) then
          if counter1 = COUNT_LIMIT1 - 1 then
             counter1 <= 0;
             temp_buzzer_1 <= not temp_buzzer_1; -- Toggle buzzer
          else
             counter1 <= counter1 + 1;
          end if;
       end if;
                              end if;
    end process;
process(clk,SW2)
    begin
              if SW2= '1' THEN
       if rising_edge(clk) then
          if counter2 = COUNT_LIMIT2 - 1 then
             counter2 <= 0;
             temp_buzzer_2 <= not temp_buzzer_2; -- Toggle buzzer
          else
             counter2 <= counter2 + 1;
          end if;
       end if;
                              end if;
```

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 6, June 2026

DOI: 10.17148/IARJSET.2026.13607

```
end process;
            process(clk,SW3)
begin
            if SW3= '1' THEN
   if rising_edge(clk) then
      if counter3 = COUNT_LIMIT3 - 1 then
         counter3 <= 0;
         temp_buzzer_3 <= not temp_buzzer_3; -- Toggle buzzer
      else
         counter3 <= counter3 + 1;
      end if;
   end if;
                        end if;
end process;
            process(clk,SW4)
begin
            if SW4= '1' THEN
   if rising_edge(clk) then
      if counter4 = COUNT_LIMIT4 - 1 then
         counter4 <= 0;
         temp_buzzer_4 <= not temp_buzzer_4; -- Toggle buzzer
      else
         counter4 <= counter4 + 1;
      end if;
   end if;
                        end if;
end process;
            process(clk,SW5)
begin
            if SW5= '1' THEN
   if rising_edge(clk) then
      if counter5 = COUNT_LIMIT5 - 1 then
         counter5 <= 0;
         temp_buzzer_5 <= not temp_buzzer_5; -- Toggle buzzer
      else
         counter5 <= counter5 + 1;
      end if;
   end if;
                        end if;
end process;
            process(clk,SW6)
begin
            if SW6= '1' THEN
                  if rising_edge(clk) then
      if counter6 = COUNT_LIMIT6 - 1 then
         counter6 <= 0;
         temp_buzzer_6 <= not temp_buzzer_6; -- Toggle buzzer
      else
         counter6 <= counter6 + 1;
      end if;
   end if;
                        end if;
end process;
            process(clk,SW7)
begin
            if SW7= '1' THEN
                  if rising_edge(clk) then
      if counter7 = COUNT_LIMIT7 - 1 then
         counter7 <= 0;
         temp_buzzer_7 <= not temp_buzzer_7; -- Toggle buzzer
      else
         counter7 <= counter7 + 1;
      end if;
   end if;
                        end if;
end process;
            process(clk,SW8)
begin
            if SW8= '1' THEN
   if rising_edge(clk) then
      if counter8 = COUNT_LIMIT8 - 1 then
         counter8 <= 0;
         temp_buzzer_8 <= not temp_buzzer_8; -- Toggle buzzer
```

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 6, June 2026

DOI: 10.17148/IARJSET.2026.13607

```
        else
            counter8 <= counter8 + 1;
        end if;
    end if;
                        end if;
end process;
            process(clk,SW9)
begin
            if SW9= '1' THEN
    if rising_edge(clk) then
        if counter9 = COUNT_LIMIT9 - 1 then
            counter9 <= 0;
            temp_buzzer_9 <= not temp_buzzer_9; -- Toggle buzzer
        else
            counter9 <= counter9 + 1;
        end if;
    end if;
                        end if;
end process;
            process(clk,SW10)
begin
            if SW10= '1' THEN
    if rising_edge(clk) then
        if counter10 = COUNT_LIMIT10 - 1 then
            counter10 <= 0;
            temp_buzzer_10 <= not temp_buzzer_10; -- Toggle buzzer
        else
            counter10 <= counter10 + 1;
        end if;
    end if;
                        end if;
end process;
            process(clk,SW11)
begin
            if SW11= '1' THEN
    if rising_edge(clk) then
        if counter11 = COUNT_LIMIT11 - 1 then
            counter11 <= 0;
            temp_buzzer_11 <= not temp_buzzer_11; -- Toggle buzzer
        else
            counter11 <= counter11 + 1;
        end if;
    end if;
                        end if;
end process;
            process(clk,SW12)
begin
            if SW12= '1' THEN
    if rising_edge(clk) then
        if counter12 = COUNT_LIMIT12 - 1 then
            counter12 <= 0;
            temp_buzzer_12 <= not temp_buzzer_12; -- Toggle buzzer
        else
            counter12 <= counter12 + 1;
        end if;
    end if;
                        end if;
end process;
            process(clk,SW13)
begin
            if SW13= '1' THEN
    if rising_edge(clk) then
        if counter13 = COUNT_LIMIT13 - 1 then
            counter13 <= 0;
            temp_buzzer_13 <= not temp_buzzer_13; -- Toggle buzzer
        else
            counter13 <= counter13 + 1;
        end if;
    end if;
                        end if;
end process;
process(clk,SW14)
begin
```

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 6, June 2026

DOI: 10.17148/IARJSET.2026.13607

```
            if SW14= '1' THEN
   if rising_edge(clk) then
      if counter14 = COUNT_LIMIT14 - 1 then
         counter14 <= 0;
         temp_buzzer_14 <= not temp_buzzer_14; -- Toggle buzzer
      else
         counter14 <= counter14 + 1;
      end if;
   end if;
                        end if;
end process;
            process(clk,SW15)
begin
            if SW15= '1' THEN
   if rising_edge(clk) then
      if counter15 = COUNT_LIMIT15 - 1 then
         counter15 <= 0;
         temp_buzzer_15 <= not temp_buzzer_15; -- Toggle buzzer
      else
         counter15 <= counter15 + 1;
      end if;
   end if;
                        end if;
end process;
            process(clk,SW16)
begin
            if SW16= '1' THEN
   if rising_edge(clk) then
      if counter16 = COUNT_LIMIT16 - 1 then
         counter16 <= 0;
         temp_buzzer_16 <= not temp_buzzer_16; -- Toggle buzzer
      else
         counter16 <= counter16 + 1;
      end if;
   end if;
                        end if;
end process;
            process(clk,SW17)
begin
            if SW17= '1' THEN
   if rising_edge(clk) then
      if counter17 = COUNT_LIMIT17 - 1 then
         counter17 <= 0;
         temp_buzzer_17 <= not temp_buzzer_17; -- Toggle buzzer
      else
         counter17 <= counter17 + 1;
      end if;
   end if;
                        end if;
end process;
            process(clk,SW18)
begin
            if SW18= '1' THEN
   if rising_edge(clk) then
      if counter18 = COUNT_LIMIT18 - 1 then
         counter18 <= 0;
         temp_buzzer_18 <= not temp_buzzer_18; -- Toggle buzzer
      else
         counter18 <= counter18 + 1;
      end if;
   end if;
                        end if;
end process;
            process(clk,SW19)
begin
            if SW19= '1' THEN
   if rising_edge(clk) then
      if counter19 = COUNT_LIMIT19 - 1 then
         counter19 <= 0;
         temp_buzzer_19 <= not temp_buzzer_19; -- Toggle buzzer
      else
         counter19 <= counter19 + 1;
      end if;
```

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 6, June 2026

DOI: 10.17148/IARJSET.2026.13607

```
    end if;
                        end if;
end process;
            process(clk,SW20)
begin
            if SW20= '1' THEN
    if rising_edge(clk) then
       if counter20 = COUNT_LIMIT20 - 1 then
          counter20 <= 0;
          temp_buzzer_20 <= not temp_buzzer_20; -- Toggle buzzer
       else
          counter20 <= counter20 + 1;
       end if;
    end if;
                        end if;
end process;
            process(clk,SW21)
begin
            if SW21= '1' THEN
    if rising_edge(clk) then
       if counter21 = COUNT_LIMIT21 - 1 then
          counter21 <= 0;
          temp_buzzer_21 <= not temp_buzzer_21; -- Toggle buzzer
       else
          counter21 <= counter21 + 1;
       end if;
    end if;
                        end if;
end process;
            process(clk,SW22)
begin
            if SW22= '1' THEN
    if rising_edge(clk) then
       if counter22 = COUNT_LIMIT22 - 1 then
          counter22 <= 0;
          temp_buzzer_22 <= not temp_buzzer_22; -- Toggle buzzer
       else
          counter22 <= counter22 + 1;
       end if;
    end if;
                        end if;
end process;
            process(clk,SW23)
begin
            if SW23= '1' THEN
    if rising_edge(clk) then
       if counter23 = COUNT_LIMIT23 - 1 then
          counter23 <= 0;
          temp_buzzer_23 <= not temp_buzzer_23; -- Toggle buzzer
       else
          counter23 <= counter23 + 1;
       end if;
    end if;
                        end if;
end process;
            process(clk,SW24)
begin
            if SW24= '1' THEN
    if rising_edge(clk) then
       if counter24 = COUNT_LIMIT24 - 1 then
          counter24 <= 0;
          temp_buzzer_24 <= not temp_buzzer_24; -- Toggle buzzer
       else
          counter24 <= counter24 + 1;
       end if;
    end if;
                        end if;
end process;
            process(clk,SW25)
begin
            if SW25= '1' THEN
    if rising_edge(clk) then
       if counter25 = COUNT_LIMIT25 - 1 then
```

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 6, June 2026

DOI: 10.17148/IARJSET.2026.13607

```
            counter25 <= 0;
            temp_buzzer_25 <= not temp_buzzer_25; -- Toggle buzzer
        else
            counter25 <= counter25 + 1;
        end if;
    end if;
                        end if;
end process;
            process(clk,SW26)
begin
            if SW26= '1' THEN
    if rising_edge(clk) then
        if counter26 = COUNT_LIMIT26 - 1 then
            counter26 <= 0;
            temp_buzzer_26 <= not temp_buzzer_26; -- Toggle buzzer
        else
            counter26 <= counter26 + 1;
        end if;
    end if;
                        end if;
end process;
            process(clk,SW27)
begin
            if SW27= '1' THEN
    if rising_edge(clk) then
        if counter27 = COUNT_LIMIT27 - 1 then
            counter27 <= 0;
            temp_buzzer_27 <= not temp_buzzer_27; -- Toggle buzzer
        else
            counter27 <= counter27 + 1;
        end if;
    end if;
                        end if;
end process;
            process(clk,SW28)
begin
            if SW28= '1' THEN
    if rising_edge(clk) then
        if counter28 = COUNT_LIMIT28 - 1 then
            counter28 <= 0;
            temp_buzzer_28 <= not temp_buzzer_28; -- Toggle buzzer
        else
            counter28 <= counter28 + 1;
        end if;
    end if;
                        end if;
end process;
            process(clk,SW29)
begin
            if SW29= '1' THEN
    if rising_edge(clk) then
        if counter29 = COUNT_LIMIT29 - 1 then
            counter29 <= 0;
            temp_buzzer_29 <= not temp_buzzer_29; -- Toggle buzzer
        else
            counter29 <= counter29 + 1;
        end if;
    end if;
                        end if;
end process;
            process(clk,SW30)
begin
            if SW30= '1' THEN
    if rising_edge(clk) then
        if counter30 = COUNT_LIMIT30 - 1 then
            counter30 <= 0;
            temp_buzzer_30 <= not temp_buzzer_30; -- Toggle buzzer
        else
            counter30 <= counter30 + 1;
        end if;
    end if;
                        end if;
end process;
```

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 6, June 2026

DOI: 10.17148/IARJSET.2026.13607

```
            process(clk,SW31)
begin
            if SW31= '1' THEN
   if rising_edge(clk) then
      if counter31 = COUNT_LIMIT31 - 1 then
         counter31 <= 0;
         temp_buzzer_31 <= not temp_buzzer_31; -- Toggle buzzer
      else
         counter31 <= counter31 + 1;
      end if;
   end if;
                        end if;
end process;
            process(clk,SW32)
begin
            if SW32= '1' THEN
   if rising_edge(clk) then
      if counter32 = COUNT_LIMIT32 - 1 then
         counter32 <= 0;
         temp_buzzer_32 <= not temp_buzzer_32; -- Toggle buzzer
      else
         counter32 <= counter32 + 1;
      end if;
   end if;
                        end if;
end process;
            process(clk,SW33)
begin
            if SW33= '1' THEN
   if rising_edge(clk) then
      if counter33 = COUNT_LIMIT33 - 1 then
         counter33 <= 0;
         temp_buzzer_33 <= not temp_buzzer_33; -- Toggle buzzer
      else
         counter33 <= counter33 + 1;
      end if;
   end if;
                        end if;
end process;
            process(clk,SW34)
begin
            if SW34= '1' THEN
   if rising_edge(clk) then
      if counter34 = COUNT_LIMIT34 - 1 then
         counter34 <= 0;
         temp_buzzer_34 <= not temp_buzzer_34; -- Toggle buzzer
      else
         counter34 <= counter34 + 1;
      end if;
   end if;
                        end if;
end process;
            process(clk,SW35)
begin
            if SW35= '1' THEN
   if rising_edge(clk) then
      if counter35 = COUNT_LIMIT35 - 1 then
         counter35 <= 0;
         temp_buzzer_35 <= not temp_buzzer_35; -- Toggle buzzer
      else
         counter35 <= counter35 + 1;
      end if;
   end if;
                        end if;
end process;
            process(clk,SW36)
begin
            if SW36= '1' THEN
   if rising_edge(clk) then
      if counter36 = COUNT_LIMIT36 - 1 then
         counter36 <= 0;
         temp_buzzer_36 <= not temp_buzzer_36; -- Toggle buzzer
      else
```

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 6, June 2026

DOI: 10.17148/IARJSET.2026.13607

```
            counter36 <= counter36 + 1;
        end if;
    end if;
                        end if;
end process;
            process(clk, SW1,SW2,SW3,SW4,SW5,SW6,SW7,SW8,SW9,SW10,SW11,SW12)
            begin
            if (SW1='1') then
            temp_buzzer1 <= temp_buzzer_1;
            elsif (SW2='1') then
            temp_buzzer1 <= temp_buzzer_2;
            elsif (SW3='1') then
            temp_buzzer1 <= temp_buzzer_3;
            elsif (SW4='1') then
            temp_buzzer1 <= temp_buzzer_4;
            elsif (SW5='1') then
            temp_buzzer1 <= temp_buzzer_5;
            elsif (SW6='1') then
            temp_buzzer1 <= temp_buzzer_6;
            elsif (SW7='1') then
            temp_buzzer1 <= temp_buzzer_7;
            elsif (SW8='1') then
            temp_buzzer1 <= temp_buzzer_8;
            elsif (SW9='1') then
            temp_buzzer1 <= temp_buzzer_9;
            elsif (SW10='1') then
            temp_buzzer1 <= temp_buzzer_10;
            elsif (SW11='1') then
            temp_buzzer1 <= temp_buzzer_11;
            elsif (SW12='1') then
            temp_buzzer1 <= temp_buzzer_12;
            end if;
            end process;

prcess(clk,SW13,SW14,SW15,SW16,SW17,SW18,SW19,SW20,SW21,SW22,SW23,SW24,SW25,SW26,SW27,SW28,SW29,SW30,SW31,SW32,S
W33,SW34,SW35,SW36)
            begin
  if (SW13='1') then
            temp_buzzer2 <= temp_buzzer_13;
            elsif (SW14='1') then
            temp_buzzer2 <= temp_buzzer_14;
            elsif (SW15='1') then
            temp_buzzer2 <= temp_buzzer_15;
            elsif (SW16='1') then
            temp_buzzer2 <= temp_buzzer_16;
            elsif (SW17='1') then
            temp_buzzer2 <= temp_buzzer_17;
            elsif (SW18='1') then
            temp_buzzer2 <= temp_buzzer_18;
            elsif (SW19='1') then
            temp_buzzer2 <= temp_buzzer_19;
            elsif (SW20='1') then
            temp_buzzer2 <= temp_buzzer_20;
            elsif (SW21='1') then
            temp_buzzer2 <= temp_buzzer_21;
            elsif (SW22='1') then
            temp_buzzer2 <= temp_buzzer_22;
            elsif (SW23='1') then
            temp_buzzer2 <= temp_buzzer_23;
            elsif (SW24='1') then
            temp_buzzer2 <= temp_buzzer_24;
            elsif (SW25='1') then
            temp_buzzer2 <= temp_buzzer_25;
            elsif (SW26='1') then
            temp_buzzer2 <= temp_buzzer_26;
            elsif (SW27='1') then
            temp_buzzer2 <= temp_buzzer_27;
            elsif (SW28='1') then
            temp_buzzer2 <= temp_buzzer_28;
            elsif (SW29='1') then
            temp_buzzer2 <= temp_buzzer_29;
            elsif (SW30='1') then
            temp_buzzer2 <= temp_buzzer_30;
```

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.066 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 6, June 2026

DOI: 10.17148/IARJSET.2026.13607

```
            elsif (SW31='1') then
            temp_buzzer2 <= temp_buzzer_31;
            elsif (SW32='1') then
            temp_buzzer2 <= temp_buzzer_32;
            elsif (SW33='1') then
            temp_buzzer2 <= temp_buzzer_33;
            elsif (SW34='1') then
            temp_buzzer2 <= temp_buzzer_34;
            elsif (SW35='1') then
            temp_buzzer2 <= temp_buzzer_35;
            elsif (SW36='1') then
            temp_buzzer2 <= temp_buzzer_36;
            end if;
            end process;
            buzzer1 <= temp_buzzer1;
          buzzer2 <= temp_buzzer2;
            process(LED1_sig,SW1,SW2,SW3,SW4,SW5,SW6,SW7,SW8,SW9,SW10,SW11,SW12)
            BEGIN
            IF (SW1='1' OR SW2='1' OR SW3='1' OR SW4='1' OR SW5='1' OR SW6='1' OR SW7='1' OR SW8='1' OR SW9='1' OR SW10='1' OR
SW11='1' OR SW12='1') THEN
            LED1_sig<='1';
            ELSE
            LED1_sig <= '0';
            END IF;
            END PROCESS;
            LED1 <= LED1_sig;
           process(LED2_sig,SW13,SW14,SW15,SW16,SW17,SW18,SW19,SW20,SW21,SW22,SW23,
         SW24,SW25,SW26,SW27,SW28,SW29,SW30,SW31,SW32,SW33,SW34,SW35,SW36)
            BEGIN
            IF (SW13='1' OR SW14='1' OR SW15='1' OR SW16='1' OR SW17='1' OR SW18='1' OR SW19='1' OR SW20='1' OR SW21='1' OR
SW22='1' OR SW23='1' OR SW24='1' OR SW25='1' OR SW26='1' OR SW27='1' OR SW28='1' OR SW29='1' OR SW30='1' OR SW31='1' OR
SW32='1' OR SW33='1' OR SW34='1' OR SW35='1' OR SW36='1') THEN
            LED2_sig<='1';
            ELSE
            LED2_sig <= '0';
            END IF;
            END PROCESS;
            LED2 <= LED2_sig;
            process(LED1_sig)
            begin
            IF LED1_sig= '1' THEN
HEX4 <= "10001110"; -- display 0 --F
HEX3 <= "10001000";  -- display 0 -- A
ELSE
HEX4 <= "11111111";  -- blank display
HEX3 <= "11111111";  -- blank display
END IF;
end process;
process(LED2_sig)
            begin
            IF LED2_sig= '1' THEN
HEX2 <= "10010010"; -- display 0 --S
HEX1 <= "11000000";  -- display 0 -- O
HEX0 <= "11000111";  -- display 0 -- L
ELSE
HEX2 <= "11111111";  -- blank display
HEX1 <= "11111111";  -- blank display
HEX0 <= "11111111";  -- blank display
END IF;
end process;
end Behavioral;
```